1988-09

Implementation of dynamic control of a single-link flexible arm using a government micro-comput

Kirkland, Michael

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

Implementation of Dynamic Control of a Single-Link Flexible Arm Using a General Micro-Computer

by

Michael Kirkland

September 1988

Thesis Advisor: L. W. Chang

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) 69	7a NAME OF MONITORING ORGANIZATION
6c ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	7b ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000
8a NAME OF FUNDING SPONSORING ORGANIZATION 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)
Implementation of Dynamic Control of a Single-Link Flexible Arm Using a General Micro-Computer

12 PERSONAL AUTHOR(S)
KIRKLAND. Michael

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's Thesis	FROM _____ TO _____	1988 September	90

16 SUPPLEMENTARY NOTATION
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the US Government.

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Flexible Manipulator, Single-Link Manipulator Control of Flexible Manipulator

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

Today's demand for a high speed, low weight and large load capable manipulator has spurred the research on flexible manipulators. This thesis centers on an implementation of dynamic control on a single-link flexible arm utilizing a general purpose micro-computer. This research also studies the dynamic behavior of the control system with a brief comparison of the derived flexible-body-model controller to a rigid-body-model controller.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL L. W. Chang	22b TELEPHONE (Include Area Code) (408)646-2632 / 22c OFFICE SYMBOL 69Ck

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

☆ U.S. Government Printing Office 1986—606-243

Implementation of Dynamic Control of a Single-Link Flexible Arm Using a Government Micro-Computer

by

Michael Kirkland
Lieutenant, United States Navy
B.S., University of Missouri, 1980

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1988

ABSTRACT

Today's demand for a high speed, low weight and large load capable manipulator has spurred the research on flexible manipulators. This thesis centers on an implementation of dynamic control on a single-link flexible arm utilizing a general purpose micro-computer. This research also studies the dynamic behavior of the control system with a brief comparison of the derived flexible-body-model controller to a rigid-body-model controller.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENT

I wish to express my deepest gratitude to my advisor, Professor Liang-Wey Chang, for his unfailing guidance and full support for this research. I also wish to express my appreciation to Tom Christian and Jim Scholfield for their advice and technical support. Last but not least, I wish to thank my wife, Helen, and my children, Ennis and Rio, for their patience and help during my studies.

I. INTRODUCTION

A. BACKGROUND

The importance of robotic manipulators to society cannot be overemphasized either presently nor in the future. With this growing dependency, greater demands are placed on manipulators to react quicker, weigh less and support greater loads. The flexible manipulator offers low power consumption, ease of transportability, reduced material requirement, lower mounting strength and rigidity requirement and lower overall cost [Ref 1].

Most work in the past have centered on rigid body manipulators until the early 1970's when flexible manipulators began to show some promise in the space industry. To meet the need of a light-weight manipulator having greater performance capabilities, certain problems must be solved in order to fully utilize the flexible manipulator. The complexity of the flexible system makes modelling the system a challenge. The model must adequately describe the system and yet it must be simple enough to implement in order to design an adequate controller for the available hardware.

The model was derived from the use of the Equivalent Rigid Link System (ERLS) first introduced by Chang [Ref 2]. The Equivalent Rigid Link System first divided the global motions into large and small motion components and then

described the kinematics of the large motion in terms of the ERLS and related the small motion kinematics relative to ERLS. Two sets of coupled, non-linear ordinary differential equations evolved from the use of finite element discretization of deformations and Lagrangian formed equations of motions. In the large motion equations, both the large and small motion variables are non-linear. In the small motion equations, the small motion variables are linear while the large motion variables remain non-linear.

Petroka [Ref 2] experimentally validated the ERLS model through the simulation of a single-link flexible arm utilizing the Continuous System Modelling Program (CSMP) on the IBM 3033 mainframe computer. A hydraulic actuated vertical motion flexible arm was designed and built. The transverse displacement of the flexible manipulator was found using a cubic shape function. With the aid of a movie camera and strain gages, Petroka was able to validate the model due to general agreement between the simulation and the actual system response.

Gannon [Ref 3] upgraded the model using a natural mode shape function and with the aid of strain gages, a potentiometer, and an accelerometer determined the tip location and dynamic behavior. Computer simulation was accomplished by the Dynamic Simulation Language (DSL) on the IBM 3033 mainframe computer. Data acquisition was performed

using a GWI Instruments MacAdios hardware and software and a Macintosh 512k computer.

Park [Ref 4] used the Dynamic Simulation Language (DSL) on the IBM 3033 mainframe computer to design a closed loop controller for the flexible arm using torque as the control variable. Using three loading conditions, Park designed a non-linear, time-variant controller. However for implementation on a common AT micro-computer, the computations must be simplified.

B. PURPOSE

The purpose of this study is the implementation of a dynamic control of a single-link flexible arm and experimentation with various parameters to study the dynamic behavior of the control system. The tip position was determined by the outputs of a potentiometer and a strain gage. Data acquisition was performed using Data Translation high speed interface board (DT 2821-F-8DI). The board supports sixteen twelve bit A/D input channels and two D/A output channels and sixteen digital input/output channels with a maximum usable sampling rate of 130 kilohertz. The micro-computer used was the standard IBM AT. The support software (AT-LAB) allowed direct manipulation of the data acquisition board through the use of provided subroutines which were compatible with FORTRAN, the language chosen to implement the controller.

The remainder of this thesis is organized into three chapters.

1. Chapter II contains the theory behind the system model.

2. Chapter III contains the design and implementation procedure of the controller.

3. Chapter IV presents the results of the closed loop system compared to the predicted values of the simulation and a brief comparison with a rigid-body-model controller. The chapter ends with conclusions and recommendations.

II. MODEL THEORY AND PLANT DESIGN

A. PHYSICAL PLANT

Petroka built a one (1) meter long flexible arm to validate the Equivalent Rigid Link System Model. The arm and the control apparatus are shown in Figures 2.1 and 2.2 respectively. The arm is flexible in the vertical plane and stiff in torsion and in the horizontal. The arm is electrohydraulically driven with a York hydraulic power unit, the pitch axis of a Berd-Johnson 3-axis Hyd-Ro-Wrist, a Moog 760-100 servovalve with its servocontroller and a high pressure filter assembly [Ref 2]. The basic construction of the arm is two thin one (1) meter parallel steel strips connected by thin steel strips to transverse steel plates. Loading is accomplished by attaching custom made .453 kg plates to the tip end transverse plate with four (4) welded studs as shown in Figure 2.3. Table 1 shows the geometric and mass properties of the flexible arm.

TABLE I. ARM GEOMETRIC AND MASS PROPERTIES

Parameter	Value	Unit
Arm length	0.9985	m
Beam thickness	0.003175	m
Beam Width	0.0762	m
Distance between each beam	0.05715	m
Arm mass	4.8565	kg
Modulus of elasticity	2.0E11	N/m^2
Density (per unit volume)	7861.05	kg/m^3

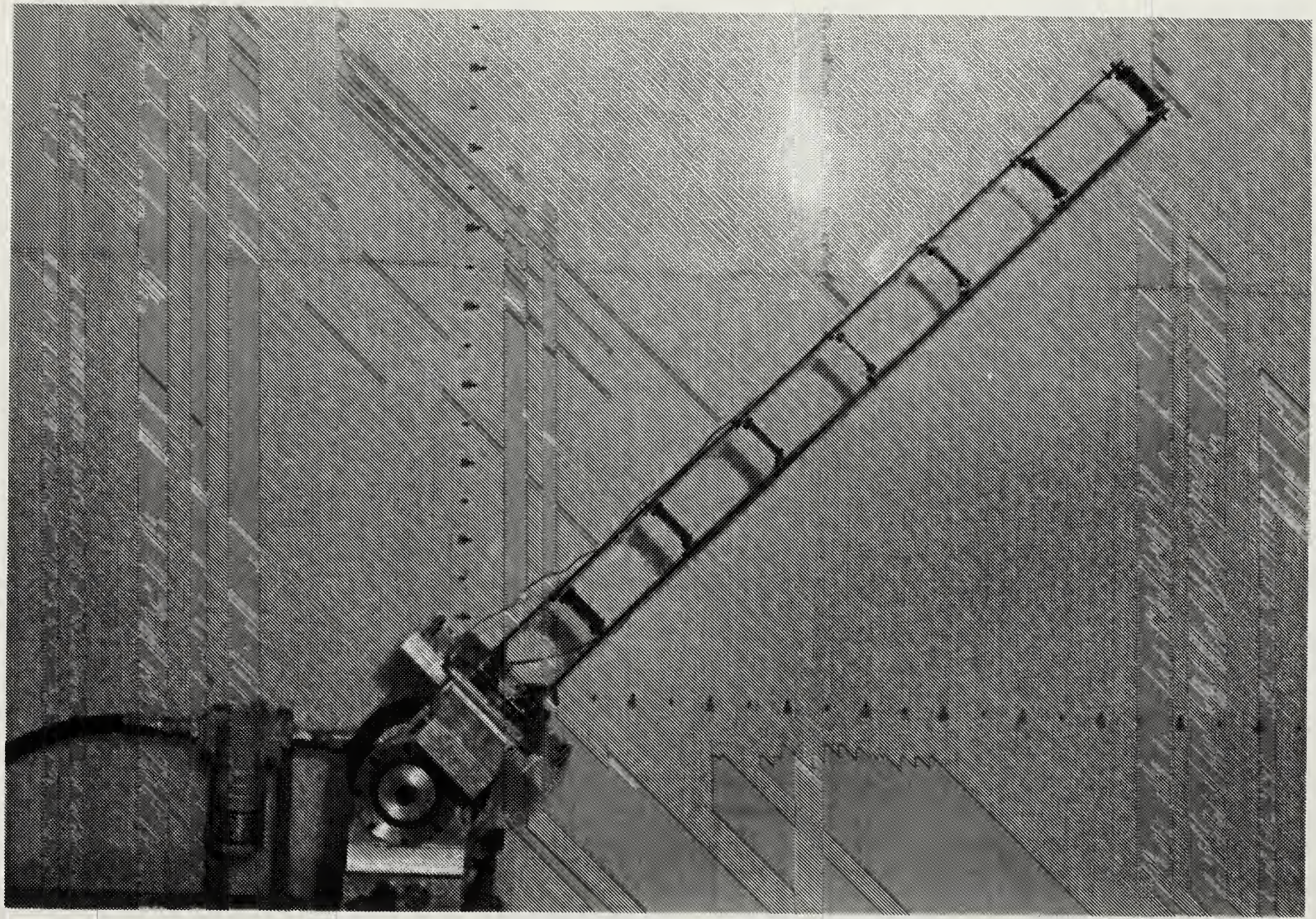

Figure 2.1 - Flexible Link, Actuator and Filter Assembly

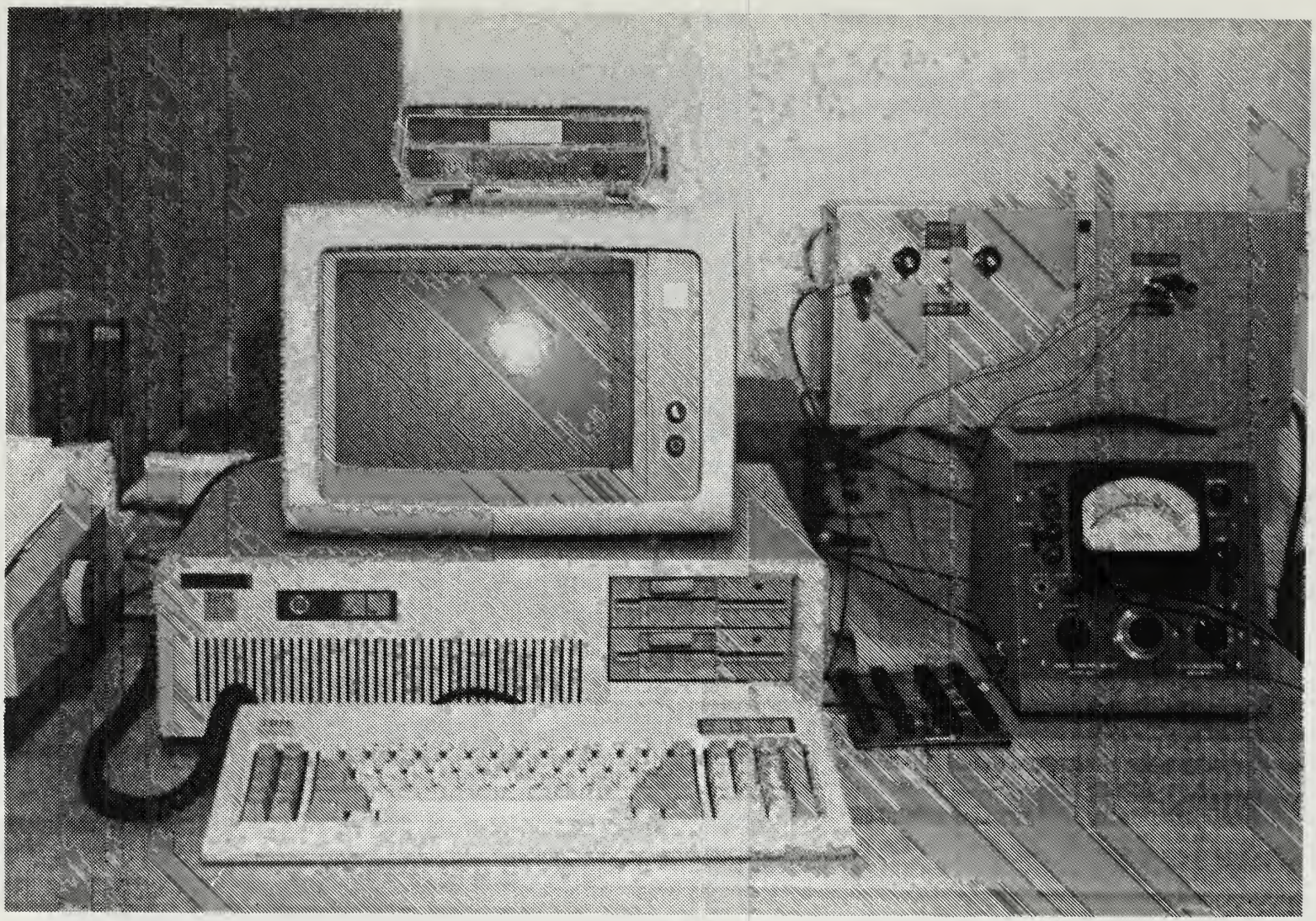

Figure 2.2 - Flexible Link Control Apparatus

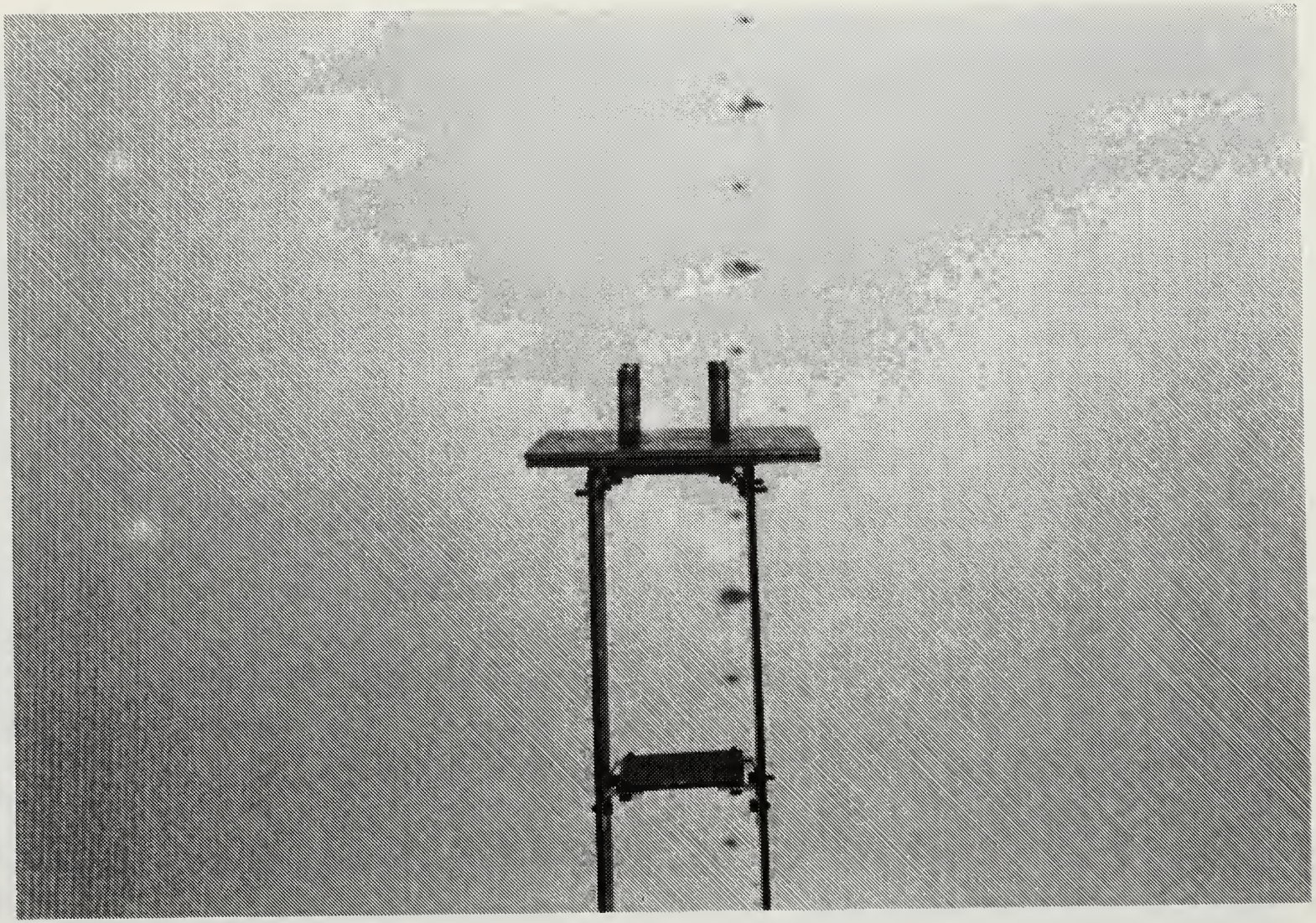

Figure 2.3 - Flexible Arm With Weights Attached

B. EQUIVALENT RIGID LINK SYSTEM FOR A SINGLE-LINK ARM

The ERLS separates the motion of the flexible link system into a large rigid body motion and a small motion displacement due to the arm's flexibility. The ERLS defines the large motion of a single-link flexible manipulator. The small motion is then described relative to the ERLS. Figure 2.4 is the schematic diagram for a single flexible link.

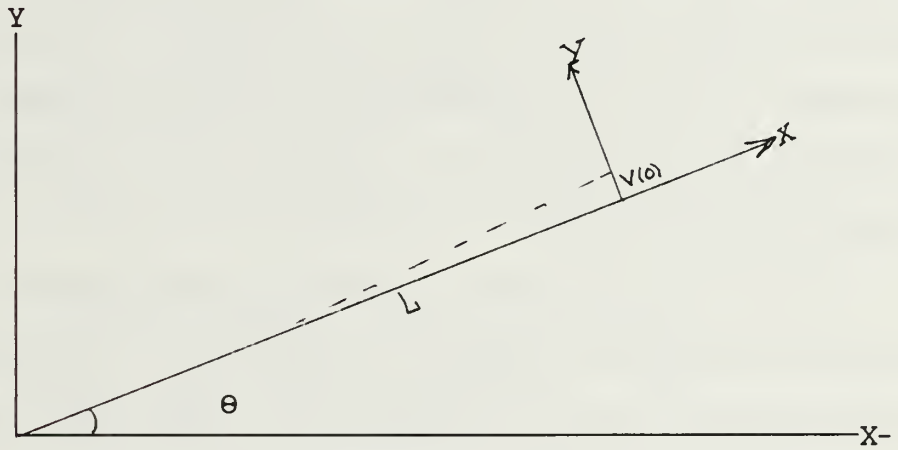

Figure 2.4

X = Inertia frame coordinate
Y = Inertia frame coordinate
x = Local frame coordinate
y = Local frame coordinate
V/L = Deflection angle
θ = Large angle

Three generalized coordinates are used to describe the large joint variable (θ), the tip deflection V(0) and tip slope $\delta(0)$. The large motion joint variable (θ), is the angle between the ERLS and the horizontal axis of the inertial coordinate system. V(0) is the tip deflection measured from the ERLS defined position. $\delta(0)$ is the tip

slope defined in terms of the local coordinates for the ERLS. Homogeneous transformations are applied to a point along the arm to obtain the point's absolute position. Since the displacements of each point along the arm is a function of location and time, the deformations must be discretized in terms of the generalized coordinates in order to take advantage of Lagrange's equation. Discretization was accomplished by the technique of the Finite Element Method (FEM) [Ref 2]. Once the kinematic relationships between the large and small motions are described, the kinetics are introduced to complete the derivation of the equations of motion.

Due to its systematic approach, the Lagrangian dynamics approach is used to derive te equations of motion. The total kinetic energy is comprised of the individual kinetic energies of the link, actuator, and any applied forces. The total potential energy of the system is comprised of the elastic strain energy of the link and the potential energy due to gravity. Generalized forces are made up of any applied forces and damping forces. Through mathematical manipulations and simplifications, two sets of coupled non-linear equations are derived. One set describes the large motion and the other set describes the small motions.

These equations describe large motion and small motion respectively;

$$M_{qq}\,\ddot{\theta} + M_{qn}\,\ddot{V} = F_q + \text{Torque} \qquad (2.1)$$

$$M_{nq}\,\ddot{\theta} + M_{nn}\,\ddot{V} + K_n\,V = F_n \qquad (2.2)$$

where

M_{qq} = 1x1 coupled effective inertia matrix for large motions.

M_{qn} = 1x2 coupled inertia matrix of the small motion effect on large motions.

M_{nq} = 2x1 coupled inertia matrix of the large motion effect on the small motions.

M_{nn} = 2x2 effective inertia matrix for small motions.

K_n = 2x2 stiffness matrix for small motions.

F = 1x1 load vector for large motions

θ = generalized coordinate of the large motions.

V = 2x1 generalized coordinates of the small motions.

The assumptions applied for this research are that;

a. Axial and torsional deformations are negligible.

b. The tip displacements are small thereby,

$$TAN^{-1}\,(V/L) = V/L \qquad (2.3)$$

with

$$V=V(O)$$

The actual slope is also assumed to be negligible. Thus, only the tip deformation V is the only generalized coordinate being used for small motion calculations. Based on these assumptions all the coefficient matrices of the non-linear, coupled, second order ordinary differential equations are reduced to 1x1 matrices. The total motion of the arm tip is represented by;

$$\Phi = \theta + V/L \tag{2.4}$$

$$\dot{\Phi} = \dot{\theta} + \dot{V}/L \tag{2.5}$$

$$\ddot{\Phi} = \ddot{\theta} + \ddot{V}/L \tag{2.6}$$

For a more detailed derivation of the equations of motions see [Ref 4].

C. SHAPE FUNCTION DERIVATION

Modelling the beam as a continuous Euler-Bernoulli cantilever beam, a natural-mode shape function was used to present the flexural motion of the single-link flexible arm [Ref 4]. The transverse displacement, u(x,t), for a single-link flexible arm is represented in the following forms;

$$u(x,t) = a_1(t)X_1(x) + a_2(t)X_2(x) \qquad (2.7)$$

$$= a_1\{A'_1\ \{\cos\beta_1 x + \cosh\beta_1 x\} + \qquad (2.8)$$

$$\{\sin\beta_1 x + \sinh\beta_1 x\}\} +$$

$$a_2\{A'_2\ \{\cos\beta_2 x + \cosh\beta_2 x\} +$$

$$\{\sin\beta_2 x + \sinh\beta_2 x\}\}$$

where

$$\beta_1 L = 1.875104069$$

$$\beta_2 L = 4.6904091133$$

$$A'_i = (\sin\beta_i L + \sinh\beta_i L)/ \qquad (2.9)$$

$$(\cos\beta_i L + \cosh\beta_i L)$$

In this research, the shape function Ω is reduced to a 3x1 matrix after having truncated the slope function. And the 3x1 displacement vector d is

$$d = \Omega * U = \begin{bmatrix} 0 \\ 0 \\ V(x) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ M \end{bmatrix} V(0) \qquad (2.10)$$

where

$$M = \{\ C_1\ \{A'_1\ \{\cos\beta_1 x + \cosh\beta_1 x\} + \{\sin\beta_1 x + \sinh\beta_1 x\}\}$$

$$+ C_2\ \{A'_2\ \{\cos\beta_2 x + \cosh\beta_2 x\} + \{\sin\beta_2 x + \sinh\beta_2 x\}\}\}$$

$$C_1 = 2\beta_2/\{4\ A'_1\ \beta_2 - 4\ A'_2\ \beta_1\ \}$$

$$C_2 = 2\beta_1/\{4\ A'_1\ \beta_2 - 4\ A'_2\ \beta_1\ \}$$

D. HYDRAULIC ACTUATION

The single-link flexible manipulator uses electro-hydraulic actuation to drive the plant. The applied current is transformed to an output torque which positions the arm.

The hydraulic dynamics is represented by the dynamics of the servovalve and actuator. The manufacturer of the servo-valve, Moog, simplified the servo-valve dynamics into a single equation (equation 2.11) [Ref 5].

$$Q=I*K*\sqrt{P_v} \tag{2.11}$$

where

Q= Flow delivered from servovalve.

I = Input current

K = Valve sizing constant, which contributes to the hydraulic system damping

P_v = Valve pressure , P_s-P_L

P_s= Supply pressure.

P_L= Load pressure.

The continuity equation supplemented by the torque output equation yields the actuator dynamics. [Ref 6]

$$Q= Dm*\dot{\theta} + Ctm*P_L + Vt*\dot{P}_L/(4\beta e) \tag{2.12}$$

$$Td= nt*P_L*Dm \tag{2.13}$$

where

Q = Flow delivered to the actuator.

$Dm*\dot{\theta}$= Flow causing actuator rotation.

βe = Effective bulk modulus of the fluid.

V_t = Total compressed volume including actuator lines and chambers.

$V_t*\dot{P}_L/(4\beta e)$ = Compressibility flow.

T_d = Torque required to overcome inertia and move the load.

$Ctm*P_L$ = Leaking flow in the actuator

Dm = Motor displacement.

n_t = Torque efficiency

P_L = Load pressure drop.

III. CONTROLLER DESIGN

A. MODEL REFINEMENT

The controller is designed and implemented on a micro-computer which had limited capabilities in both computation speed and power. This suggests that any simplifications or refinements to the proposed non-linear, time variant, coupled equations of motion would be beneficial to the successs of implementing a controller. Without the simplifications more calculations would be needed to compute the coefficients which are used to design the controller. This would increase the time needed between system updates in order to compute and finally transmit a control signal. If the control signal is delayed too long the controller may not be able to keep up with the system.

First, the coefficients in the equations of motion were evaluated by running the simulation program developed by Park [Ref 4]. It was discovered that over a large range of values for the total angle (Φ) (from ϕ to 1 radian), that the coefficients could be considered either constant or to be a direct function of the large angle. This greatly simplified the equations from being non-linear to becoming linear equations. And due to the constancy of the coefficients, the equations become time invariant. This simplification suggested using the State Space Method for designing the controller.

Beginning with the system equations 2.1 and 2.2 and eliminating $\ddot{V}$;

$$M_{qq}\ddot{\Phi} + D(F_n - M_{nq}\ddot{\Phi} - K_n V) - F_q = \text{Torque} \qquad (3.1)$$

where

$$D=(M_{qn} - M_{qq}/L)/(M_{nn} - M_{nq}/L) \qquad (3.2)$$

Now combining terms results in a short form equation;

$$N\ddot{\Phi} + Fc(\Phi,\dot{\Phi},V,\dot{V}) = \text{Torque} \qquad (3.3)$$

where

$$N = M_{qq} - D*M_{nq} \qquad (3.4)$$

$$Fc(\Phi,\dot{\Phi},V,\dot{V}) = D*F_n - F_q - D*K_n*V \qquad (3.5)$$

N was found to approximate a constant value while Fc was found to approximate a linear function of the total angle (Φ), Figures 3.1 and 3.2 respectively. Equation 3.5 was converted using these approximations into Equation 3.6;

$$N*\ddot{\Phi} + E*\Phi = \text{Torque} - F \qquad (3.6)$$

with

E,F = Constant coefficients

Table II displays the coefficients for the two trial loading conditions.

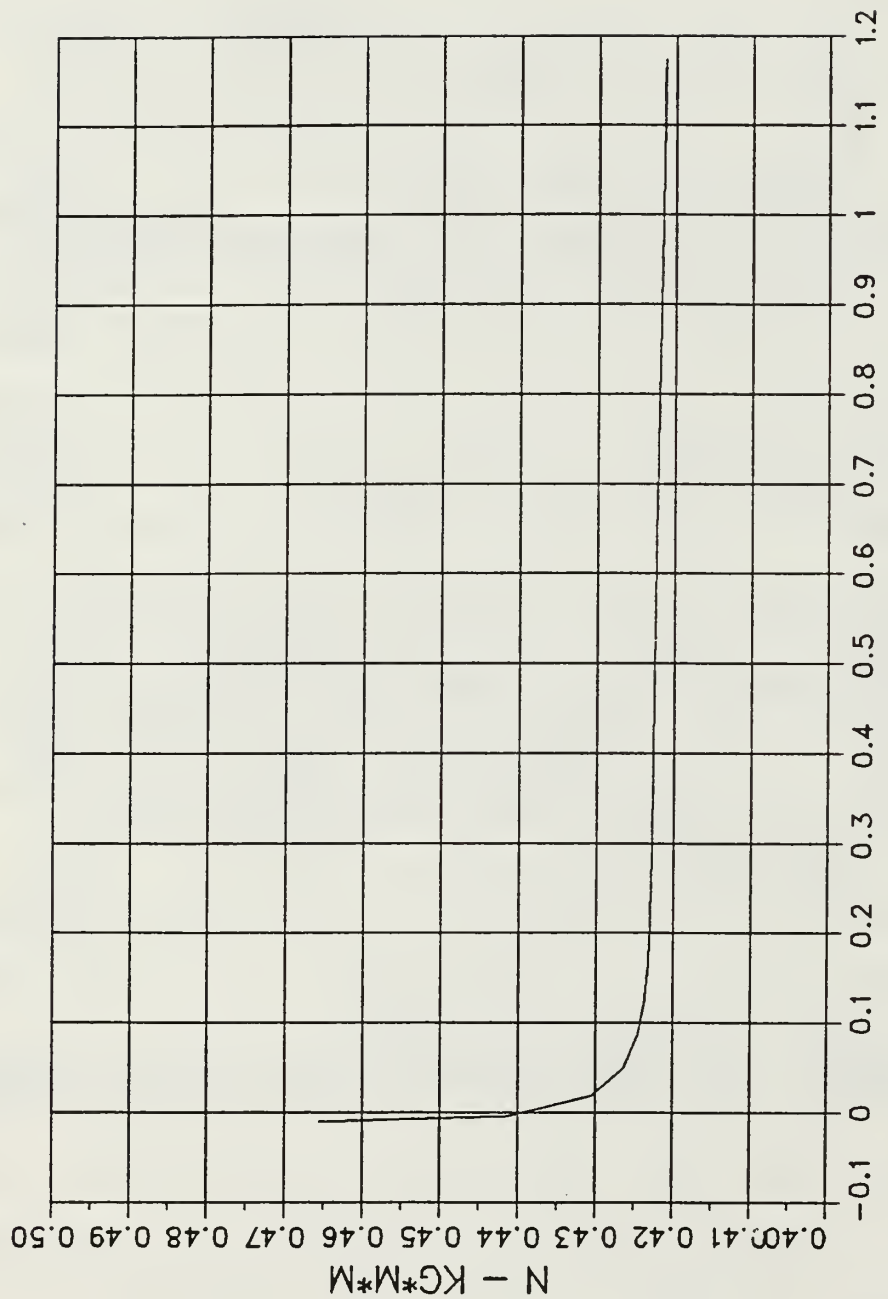

Figure 3.1 - 0 Load Inertia Coefficient N

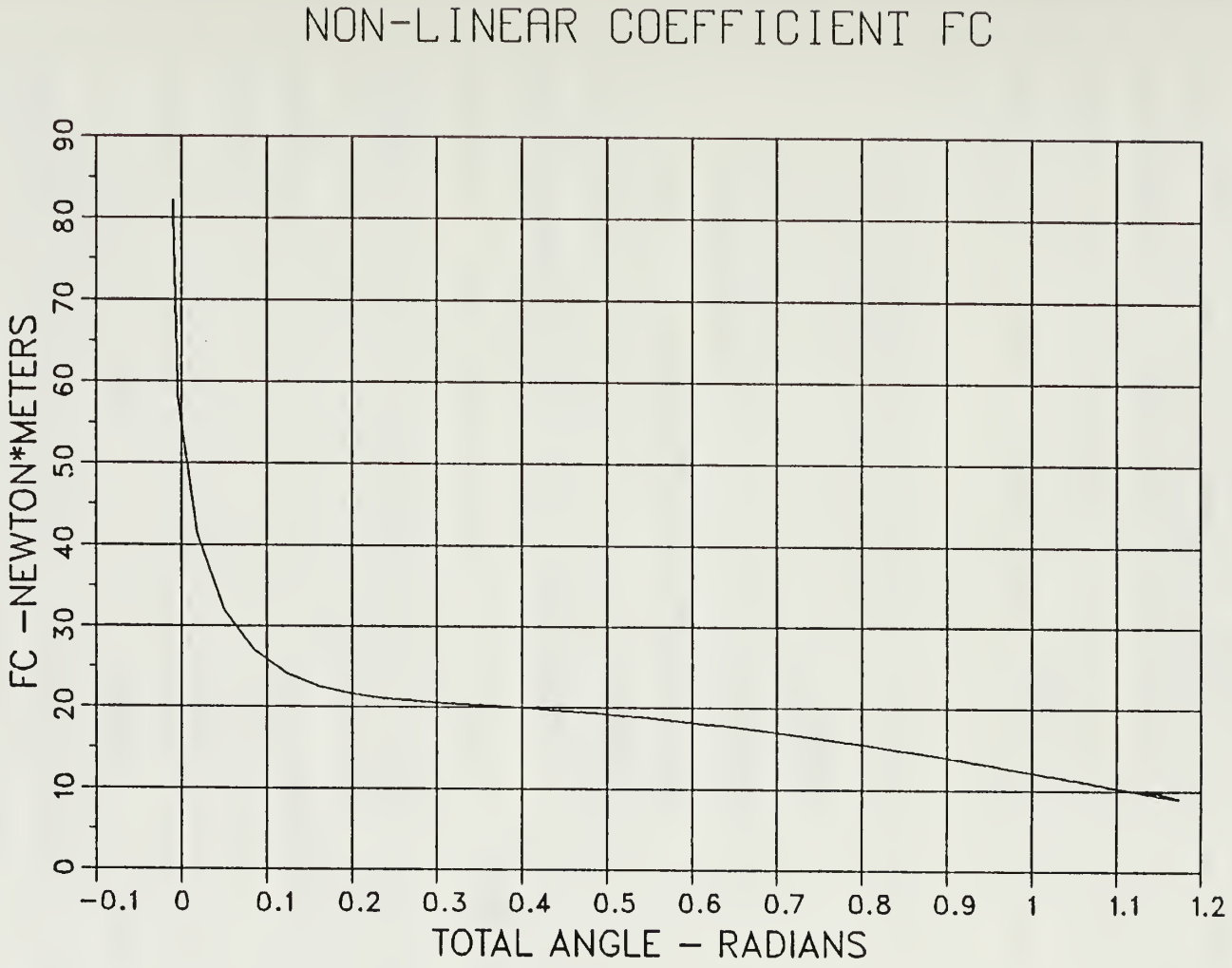

Figure 3.2 - 0 Load Non-Linear Term-Fc

TABLE II. TRIAL MASS COEFFICIENTS

MASS (kg)	N N*m*s/rad)	E (N*m/rad)	F (N*m)
0	.42164	-15.53159	28.4392
1.3593	1.2343	-17.72896	39.3924

Equation 3.6 was then transformed into the State Space variable format of

$$X = A*X + B*U \tag{3.7}$$

$$Y = C*X \tag{3.8}$$

with

U = Torque - F = The control input.

A = 2x2 Matrix

B = 2x1 Matrix

X = 2x1 State Variable Matrix

C = 1x2 Matrix

Table III has the values of the coefficient matrices for each trial mass.

TABLE III. STATE SPACE CONTINUOUS TIME COEFFICIENTS

MASS	A	B	C
0	0 1 36.8361 0	0 2.3717	1 0
1.3593 (kg)	0 1 14.3642 0	0 .8102	1 0

The problem for the purposes of controller design has been converted to a linear, time invariant problem.

B. SELECTION OF A SAMPLING RATE

The selection of the proper sampling rate greatly influences whether a successful controller can be designed. Sampling is necessary because input values are obtained by the computer at specified times either as functions of the computer clock or a function of the control program. Some of the considerations when selecting a sampling rate are;

1. The minimum time required for the D/A and A/D conversions.
2. The amount of control effort desired.
3. The required time to complete all control computations

The sampling rate must not be too low for it may give rise to aliasing or frequency folding. Aliasing and frequency folding describe the same phenomenon. This occurs if the sampling rate is not greater than twice the maximum system frequency. The frequencies which are greater than half the sampling frequency then "fold" upon themselves becoming additive in nature giving rise to an enhanced signal. If an arbitrarily high sample rate is chosen then the physical limitations imposed by the micro-computer and data acquisition hardware must be considered. Therefore, a trade-off is normally made. Shannon's Sampling Theorem states that any signal whose highest frequency F_p can be reconstructed if the sampling rate is twice F_p.

However, for control purposes, this sampling rate is generally too low. Astrom and Wittenmark suggested using six (6) to ten (10) times the system frequency bandwidth [Ref 8]. Initially, the sampling period was chosen as 0.005 seconds of five (5) milli-seconds because it was the fastest rate in which all necessary conversions and calculations could be accomplished in one sample period.

The importance of the sampling rate cannot be over-stressed. A high sampling rate requires greater control actions which is translated to greater component action that can lead to early equipment failure. A high sample rate also increases the chances of exciting higher order unmodelled system dynamics. So it is very important to understand the system under consideration in order to decide how much control is enough and the final choice of a sampling rate should be the result of that decision.

C. CONTROLLER DESIGN

The controller was designed using the State Space Design Method coupled with pole placement. The poles were selected in the continuous time S-domain and converted to the discrete time Z-domain using;

$$pd = \exp(h*p) \tag{3.9}$$

where

pd = The Z-domain poles.

p = The S-domain poles.

h = The sampling period.

The continuous time model Equation 3.7 was then converted to the discrete time difference Equation 3.10.

$$X(h*k + h) = \alpha*X(h*k) + \Gamma*U \tag{3.10}$$

where

$$\alpha = \exp(A*h) \tag{3.11}$$

$$\Gamma = \int \exp(A*h)dt * B \tag{3.12}$$

Table IV contains the coefficient matrices for the trial masses.

TABLE IV. STATE SPACE DISCRETE TIME MATRIX COEFFICIENTS

MASS	α	Γ	C
0	0.9995 0.0050 - 0.1843 0.9995	0 0.0119	1 0
1.3593 (kg)	1.0002 0.0050 0.0718 1.0002	0 0.0041	1 0

Using Equations 3.9, 3.10 and Ackerman's Equation, a 1x2 matrix of gains K was derived for given pole locations. From each set of gains, K, the control law was defined as

$$U(k) = R(k) - K*X(k) \tag{3.13}$$

where

R(k) = The reference signal.

K = 1x2 matrix = [Kp Kv]

Kp = The positional gain.

Kv = The velocity gain.

The reference input is composed of the dynamics of the system, the effects of the zero order hold, the sampling

rate and the required state. The derivation of the reference input follows.

Within some finite number of time steps(k) a system will reach its required states if the poles and gains are properly chosen assuming that the system is controllable. The at time step (k+1), the state at that time will equal the state at time(k) multiplied times a time step increment or in other words

$$zX(z) = Z(X(k+1)) \tag{3.14}$$

Equation 3.14 is the Z-transform representation. Taking the Z-transform of Equation 3.10 and replacing the left hand side with the left hand side of Equation 3.14 will yield

$$zX(z) = \alpha X(z) + \Gamma U(z) \tag{3.15}$$

The next step is taking the Z-transform of Equation 3.13 and replacing U(z) in Equation 3.15 with the right hand side of the transformed Equation 3.13.

$$zX(z) = \alpha X(z) + \Gamma (Rk(z) - KX(z)) \tag{3.16}$$

Combining terms and solving for X yields Equation 3.17.

$$X(z) = [Z*I-(\alpha-\Gamma*K)]^{-1} * \Gamma*Rk(z) \tag{3.17}$$

where

I = The identity matrix

The system output is represented by Equation 3.18.

$$Y(z) = C*X(z) \tag{3.18}$$

The relationship between the output and the reference is derived by replacing X(z) in Equation 3.18 with the results of Equation 3.17.

$$Y(z) = C*[Z*I-(\alpha - \Gamma*K)]-1 * \Gamma*Rk(z) \quad (3.19)$$

Let

$$Rk(z) = (C*[Z*I - (\alpha - \Gamma*K)]^{-1} *Y_r \quad (3.20)$$

or

$$Rk(z) = H(z)^{-1} * Y_r \quad (3.21)$$

with

Y_r = The required output.

$$H(z) = (C*]Z*I - (\alpha - \Gamma*K)]) \quad (3.22)$$

Now replace

$$Y(z) = Y_r \quad (3.23)$$

or

$$Y(z) - Y_r = 0 \quad (3.24)$$

Equation 3.24 assumes that the states are fully known and the model exactly represents the system. In actual practice Equation 3.24 doesn't equal zero but instead some small error. However, if the gains are carefully chosen and the model is a good approximation of the system, the error will be very small or could be approximated as zero.

H(z) is called the "pulse transfer function" [Ref 8]. It is normally a quotient of two polynomials in z. Even if this function is stable and causal, the inverse of the function may have problems with causality or with stability. The causality is caused by the numerator having higher order terms than the denominator. This then requires that future

information be known regarding the future output of the system to determine the present input. The problem with stability may occur if there is a pole that is ≥ 1. The problem with causality isn't seen since the output chosen is the final value for a given trajectory. The problem with stability can be alleviated by selecting a filter which replaces the unstable pole with stable pole.

Once the reference signal is computed in the complex frequency domain, it is a simple matter to transforming back to the discrete time domain by taking the inverse Z-transform of Equation 3.20. After computing H(z) the plant's trajectory can be contlled by choosing an appropriate form of Y_r, ie, constant value, linear to produce a ramp appearance, or a quadratic to produce a horseshoe effect. In this research the values of Y_r was either a constant or linear.

Once the control law was established, the control signal was determined and converted to a required torque. Since torque could not be transmitted directly, the equivalent current was generated using the inverse dynamics relations of the hydraulic actuator as described by Equations 2.12 and

$$P_L = T_d/n_t * Dm \tag{3.25}$$

$$I = Q/(K*\sqrt{P_v}) \tag{3.26}$$

Knowing the resistance allowed the computation of an output voltage which would produce the required current to actuate the actuator's torque motor. In this control

scheme, the value of current was limited to 4 mA to avoid saturation of the hydraulic actuator controller. The derived controller was then installed in the simulation model and tested to insure that it could in fact control the modelled system. This procedure was repeated until an adequate controller was found. The values of the gains and reference signals are found in Table V.

TABLE V. TABLE OF REFERENCE AND GAIN INPUTS

MASS (kg)	REFERENCE	Kp	Kv	POLES
0	867.2958	847.7606	60.00	-200.81,-16.99
1.3593	252.5000	250.4961	37.00	-22.6, 9.1

D. IMPLEMENTATION

The implementation involved the connection of the controller to the flexible arm. The full controlling system consisted of a 512k IBM micro-computer, a Bam-1 amplifier unit, a strain gage, a potentiometer, a control unit for the electrohydraulic actuator, and a high speed data acquisition system. The output from the strain gage was fed to the Bam-1 for amplification and then sent to the interface board. This arrangement provided the input for the small motion V/L.

The output from the potentiometer was fed to the actuator control box which had a built-in filter to attenuate any noise on the large motion signal. This signal was then fed to the interface board and this provided the large angle input.

The angular velocity of the total angle was determined using a reduced order observer of the form

$$X(k+1)=(X(k+1)-X(k))/h + (h/2)U(k) \qquad (3.27)$$

with

h = sampling period

U(k) = control input

The interface board was the Data Translation Acquisition Board DT-2821-F-8DI. The board supports sixteen digital input/output (I/O) channels, sixteen twelve bit analog to digital (A/D) input channels and two twelve bit digital to analog (D/A) output channels. The analog channels are capable of being configured as unipolar, or bipolar, single end or differential. The maximum useful sampling rate is 130 kilohertz.

The control sequence was then measured to ensure that it was less than the sampling rate. Three basic programs were used to design the controller and then to control the plant. The program PT0M DSL (Appendix A) was used to run simulations of the system. The program FPOINT.FOR (Appendix B) was used to establish point to point control. The program TRAJ.FOR (Appendix C) was used to establish trajectory control.

IV. RESULTS AND CONCLUSIONS

A. GENERAL COMMENTS

As part of the research, a sampling rate, which could provide the requisite control yet would minimize the control effort without attempting to design an optimal controller,is to be determined. The adopted method was to use the highest sampling rate feasible to design the controller which would require the higher gains and then to adjust the sampling rate downward until the lowest value of the required torque was achieved while retaining the control accuracy. Also it was desired to stay within the band of the suggested sampling rate of six to ten times a desired system frequency. The desired system frequency was between two to three times the arm's natural frequency to ensure a good response. This was accomplished in the design process by the choice of poles selected. It was found that at 50 Hz, the control effort or torque was cut approximately by 1/3 of the value required at 200 Hz for a zero loading condition as noted in Figures 4.1 and 4.2. Base on this, all further trials were conducted at 50 Hz. However, it was found later that in order to maintain control of the 1.36 kg load during a series of ramps and step inputs it was necessary to increase the sampling rate to 100 Hz.

During the execution of the testing, it was noticed that the arm would begin to vibrate with increasing amplitude

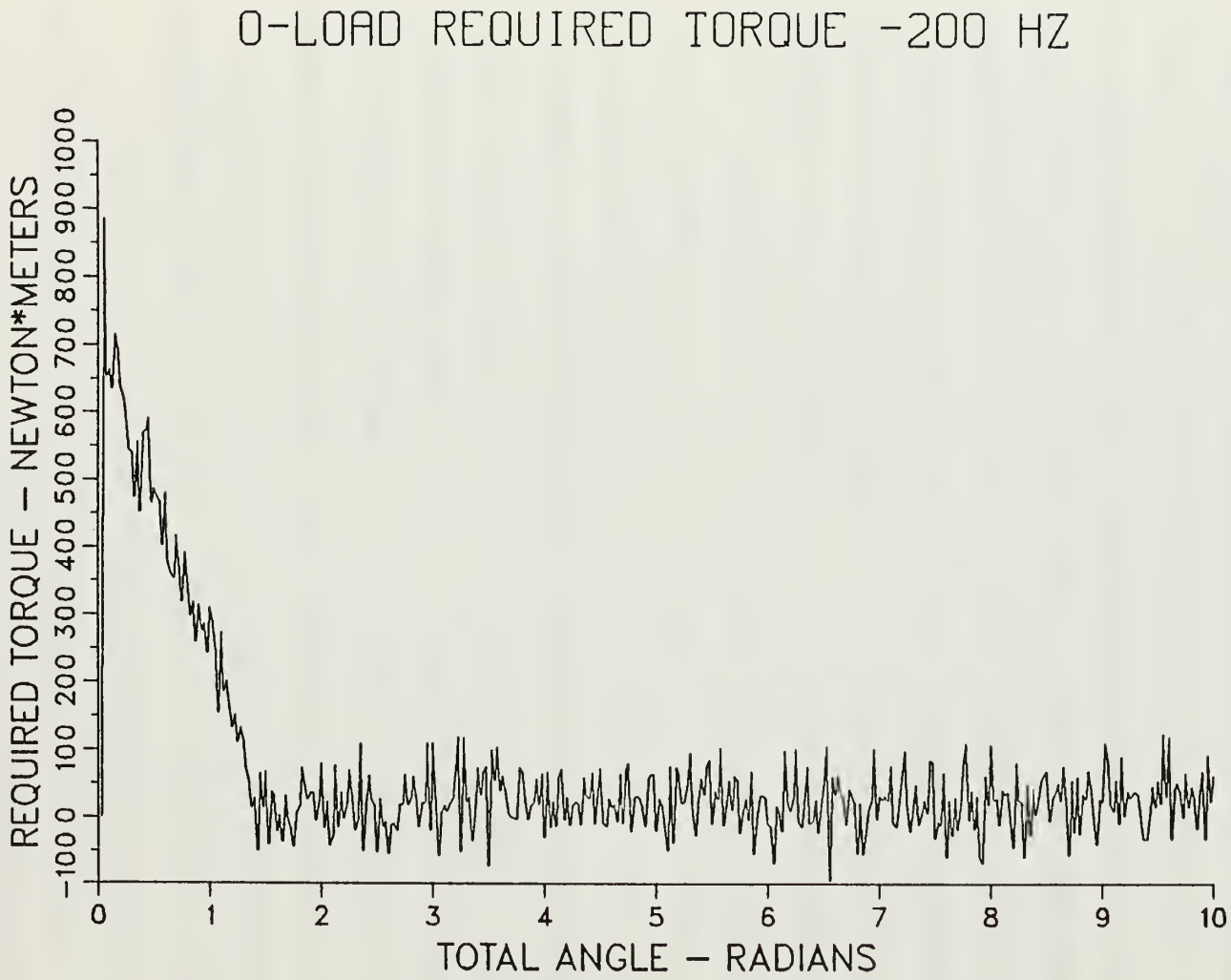

Figure 4.1 - 0 Load Required Torque At 200 Hz

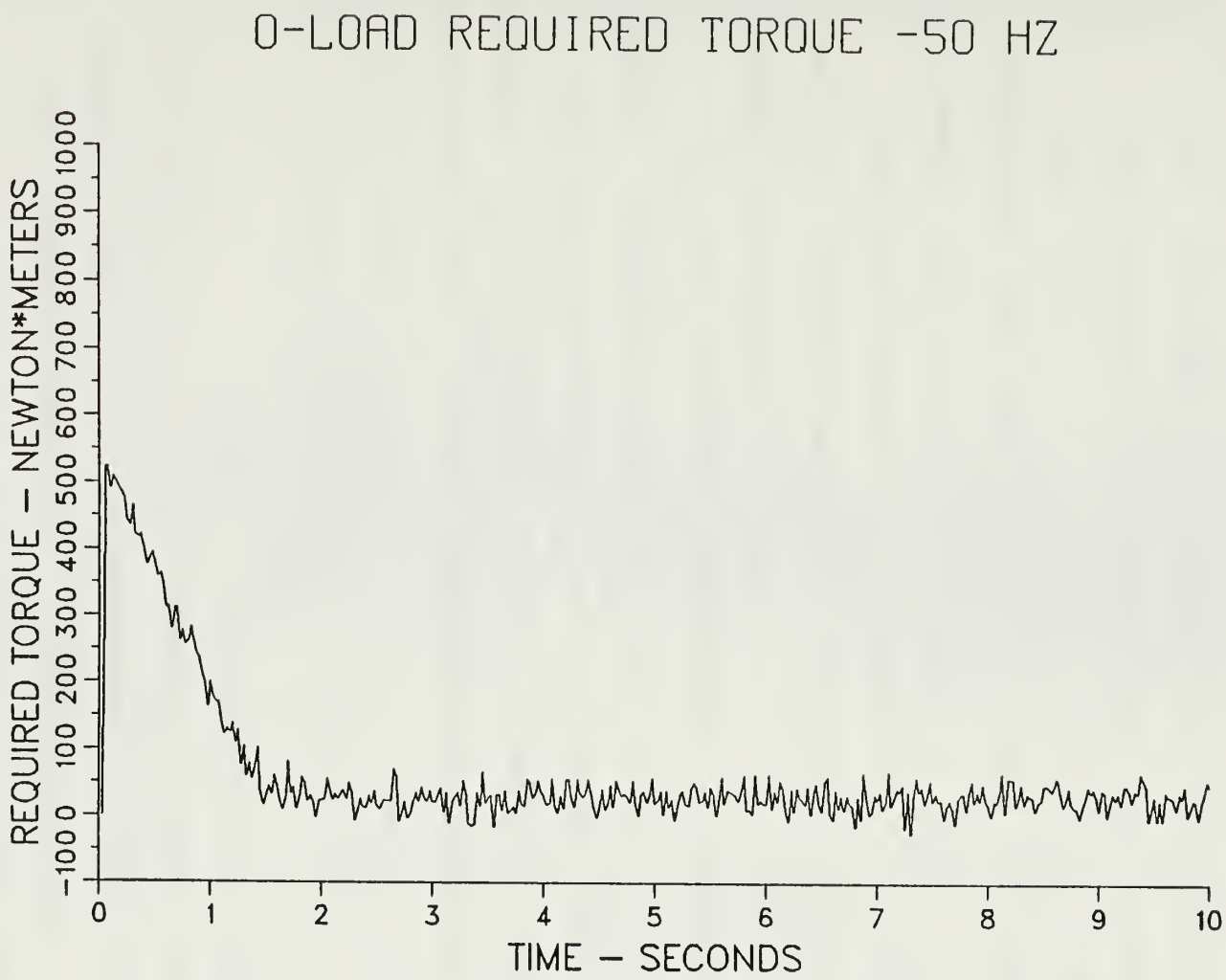

Figure 4.2 - 0 Load Required Torque At 50 Hz

after the arm had reached its required position. After analysis, it was found that since the small angle signal was of such small magnitude (>0.1 volt), that the signal was dominated by noise with frequencies above 3 Hz. This problem was alleviated by installing a digital filter designed using Tustin's bilinear rule [Ref 8]. This is where the true power of the Equivalent Rigid Link System came into play. It allowed the control of the system while the search for the proper cutoff frequency was being conducted. It basically allowed the isolation of the small angle from the control circuitry and easily facilitated the monitoring of the signal using the control program. Initially 10 Hz was tried as the cutoff frequency, however noise still dominated the signal and at some point the amplitude would begin to grow. The arm's natural frequency was computed as approximately 2 Hz and a cutoff frequency of 3 Hz was selected. An example of a dirty signal is shown in Figure 4.3 and a "clean" signal is shown in Figure 4.4.

B. POINT TO POINT CONTROL

This scenario has the arm follow a travel command from a starting point to a finishing point via response to a step input. During this scenario the effect of varying the sampling rates and control gains were tested on the zero load case while expecting similar results at higher weights only possibly amplified. The selected sampling rates used

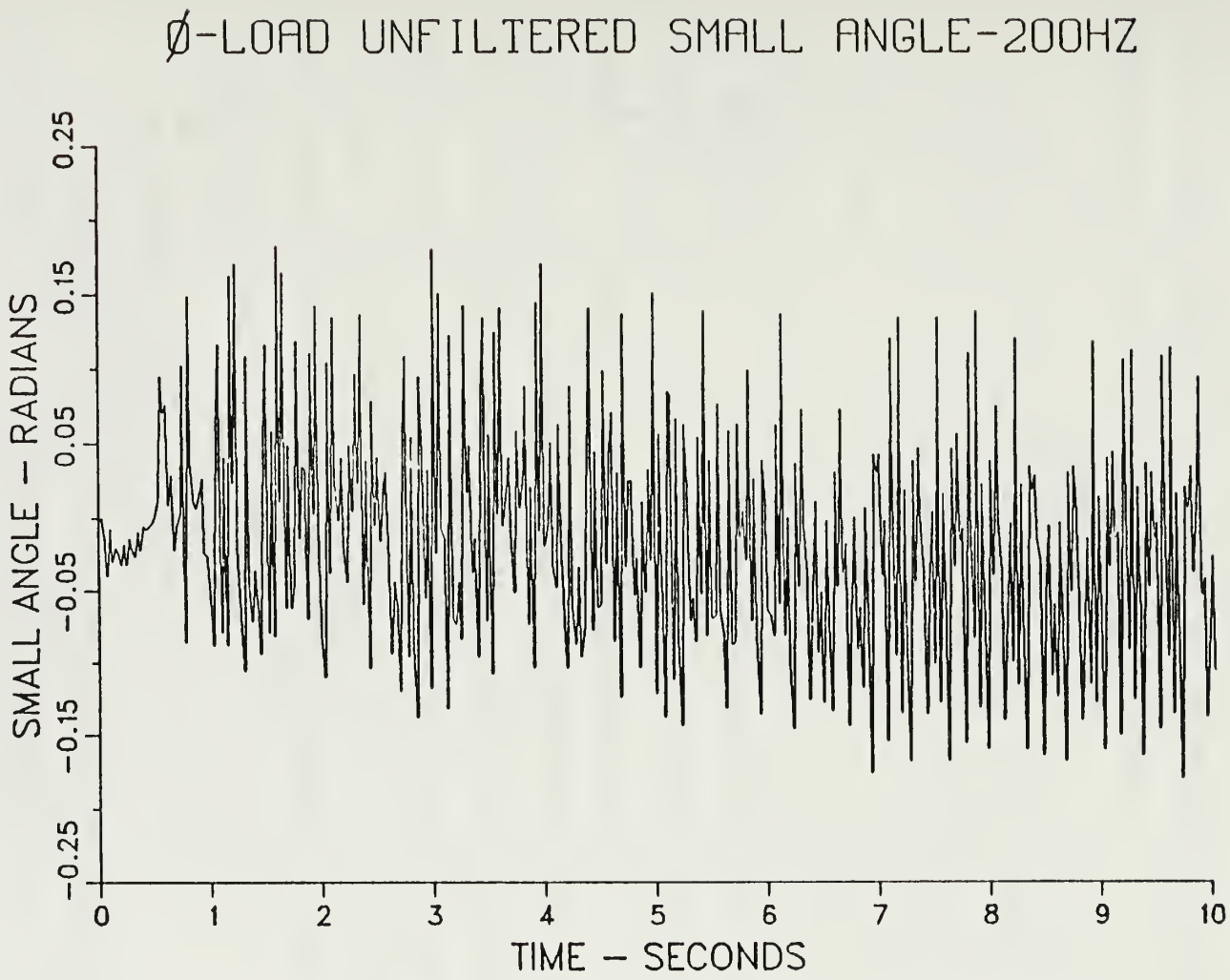

Figure 4.3 - Ø Load Unfiltered Small Angle At 200Hz

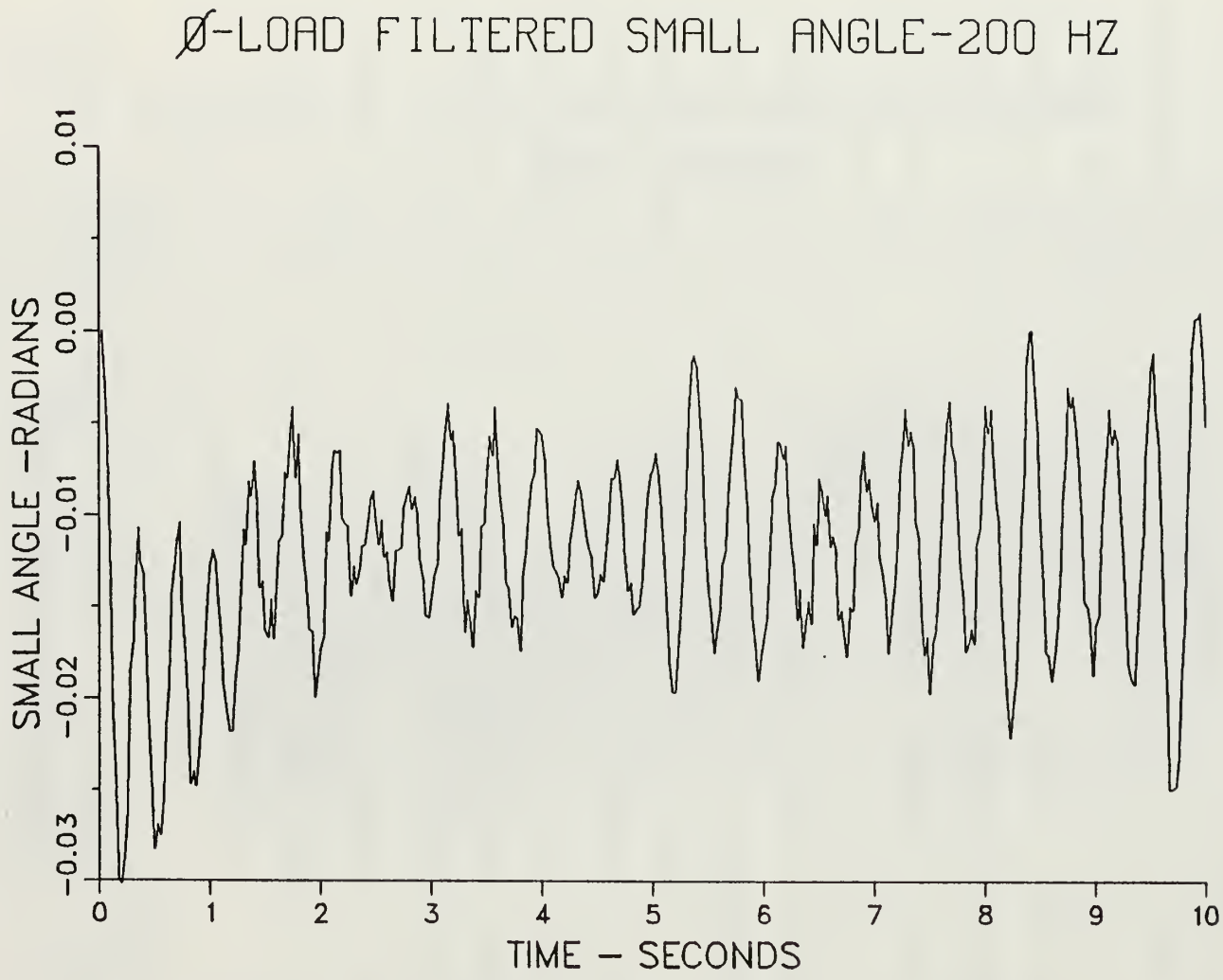

Figure 4.4 - Ø Load Filtered Small Angle Input

were 200 Hz, 100 Hz and 50 Hz along with two different sets of gain. The required output in all testing was one radian in order to remain consistant with previous research. The value of the gains and S-domain poles are in Table VI. Both sets of gains produce the same trend of the effect which sampling rate has on the steady state error.

TABLE VI. GAINS

SET	KP	KV	POLES
1	845.76	60.00	-200.8,-16.9
2	411.8	33.00	-42.4,-28.6

Figures 4.5 through 4.7 show the results generated by using the gains of set 1. It is seen that for the higher sampling rates small amplitude cycling about the final position occurs. It appears that while the higher modes of vibrations has been successfully filtered out their coupling with the fundamental mode is being activated at the higher sampling rates. However, the effect is even more pronounced at higher loads as demonstrated by Figure 4.8. It appears that lower frequency model coupling excitations occur at the loaded conditions.

C. TRAJECTORY CONTROL

In trajectory control the same gains and reference signals were used as in the point to point control portion of the research. What has been added is that the inputs are

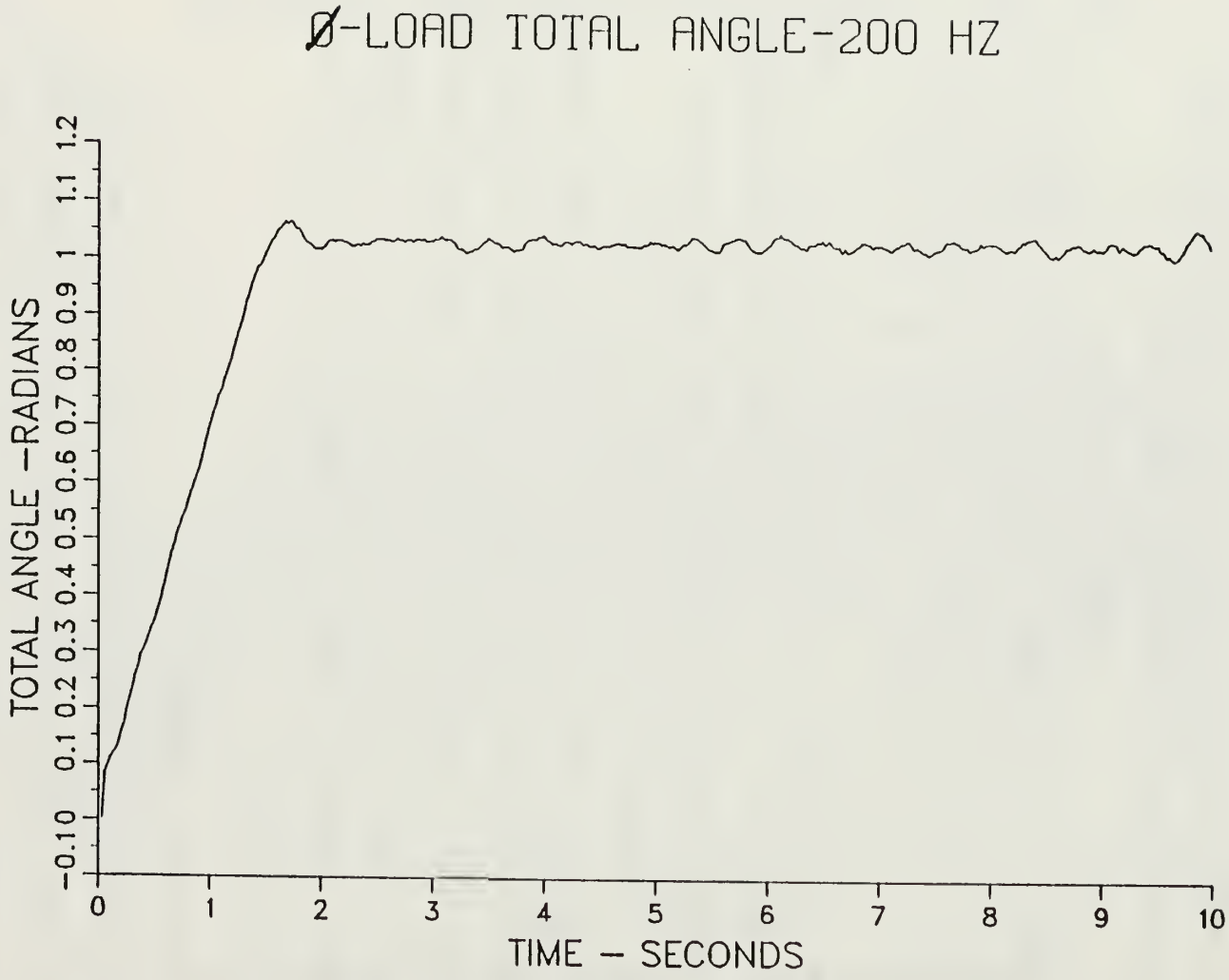

Figure 4.5 - Comparison Of Sampling Rate Effects at ø Load

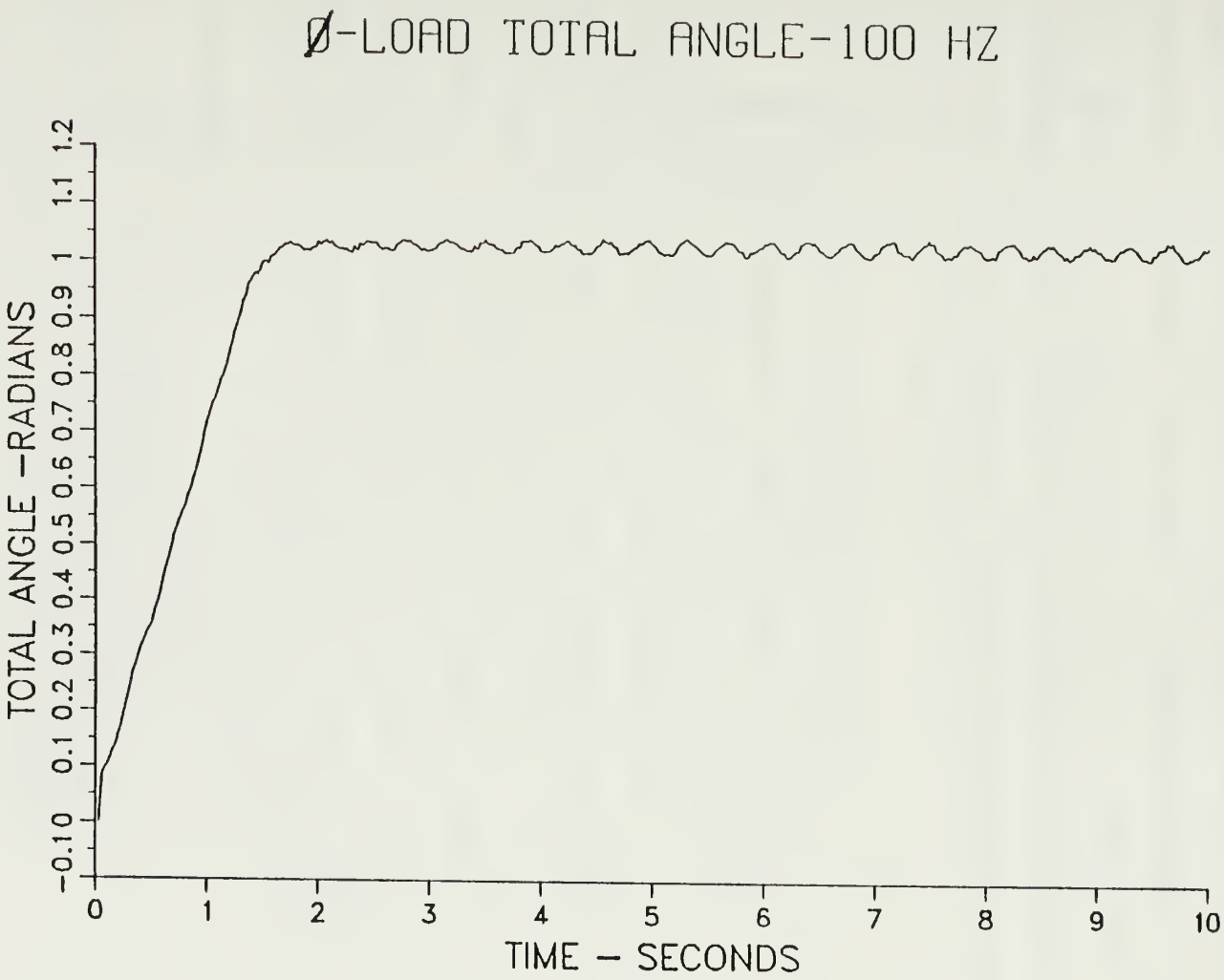

Figure 4.6 - Comparison Of Sampling Rate Effects at Ø Load

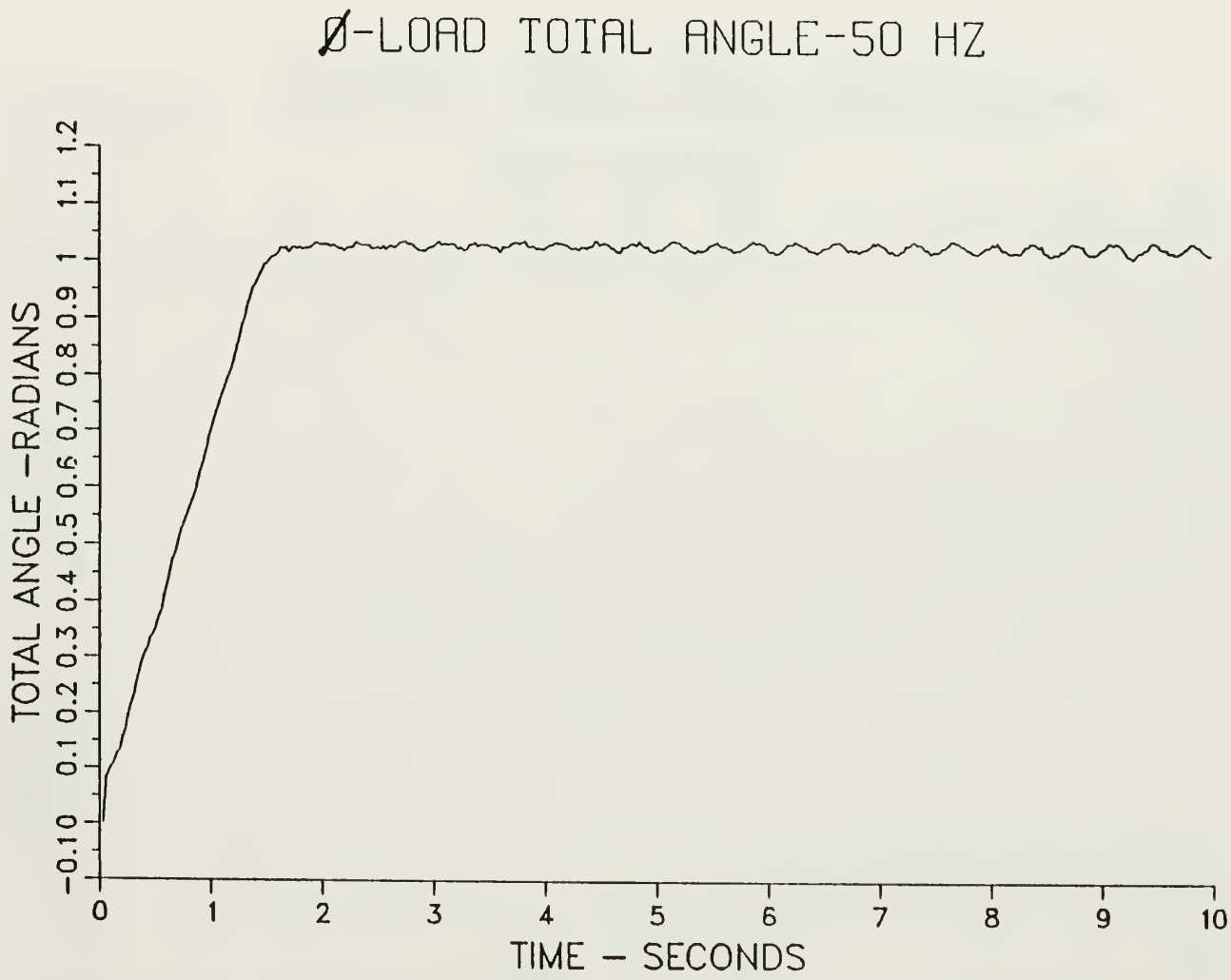

Figure 4.7 - Comparison Of Sampling Rate Effects at Ø Load

of the research. What has been added is that the inputs are a combination of ramp and step inputs. What is seen immediately is that for the zero loading case the simulated value and the actual value was in close agreement as shown in Figure 4.9. In both loading conditions the plant responded to the commands and followed the trajection with the 1.36 kg load enjoying a slightly better steady state error as seen in Figure 4.10.

D. THE PAYOFF OF THE FLEXIBLE-BODY-MODEL

Previous research suggested that the flexible-body-model would allow higher speeds, larger loads, and smaller required torques for a flexible manipulator. A modest attempt has been made to see if these advantages are realized with a single-link manipulator. Using the flexible-body-controller and designing a controller using rigid body assumptions, their performance were compared in the areas of steady state error and required torques at 0 and 1.36 Kg loads. The design of the rigid-body-controller follows.

$$I\ddot{\theta} + mgl\sin(\theta)/2 = \text{Torque} \qquad (4.1)$$

and

$$I = I_r + I_a + I_l \qquad (4.2)$$

where

I_r = The moment of inertia of the rotor.

I_a = The moment of inertia of the arm.

I_L = The moment of inertia of the load.

The controller equation for point to point control became

$$\text{Torque}=I(-k_v \dot{\theta} -k_p (\theta-Y_r))+ \quad (4.3)$$

$$mgl\sin(\theta)/2$$

where

m = Mass of the arm and load.

g = Gravitational constant.

L = Length of the arm.

Y_r = The required output.

It is seen that the steady state errors are comparable in Figure 4.11 for the ∅ load condition. Also notice that the flexible-body-model control displayed a faster settling time. However, there is clear evidence that the maximum required torque is approximately 1/3 lower for the flexible-body-model as seen in Figure 4.12. The comparison of the steady state error at the 1.36 Kg loading shows clearly that for the same set of poles with equivalent gains that the rigid-body- model is totally inadequate for that loading condition as shown in Figure 4.13. The comparison of required torque at 1.36 Kg loading again shows that the rigid body model requires much more torque as shown in Figure 4.14.

E. CONCLUSIONS

The system is highly sensitive to noise. The entire system, arm controller and actuator must be isolated from both input and output noises. Some methods for accomplishing noise isolations are;

1. Use shielded cabling.
2. Install pre-filters.
3. Measurement equipment and computer should be moved as far as possible from the hydraulic pump.

The system is highly reactive to mode coupling at higher sampling rates.

As predicted by control theory steady state error decreased with increasing gains.

The operating frequency of the system was between 1.5 to 4 times the natural frequency of the arm. The bandwidth explored for the 0 load case was from 5.6 Hz to 9.5 Hz. The bandwidth for the 1.36 Kg was 2.3 Hz.

The results of the comparison between the rigid body mode controller and the flexible body model controller are;

1. With increasing loads and at equivalent gains, the rigid-body-model has an increasing steady state error.

2. The rigid-body-model required much more torque than the flexible-body-model without providing greater control.

It appears that the results obtain points toward the results of the previous research. However, caution must be exercised and more comparison should be sought at greater loads.

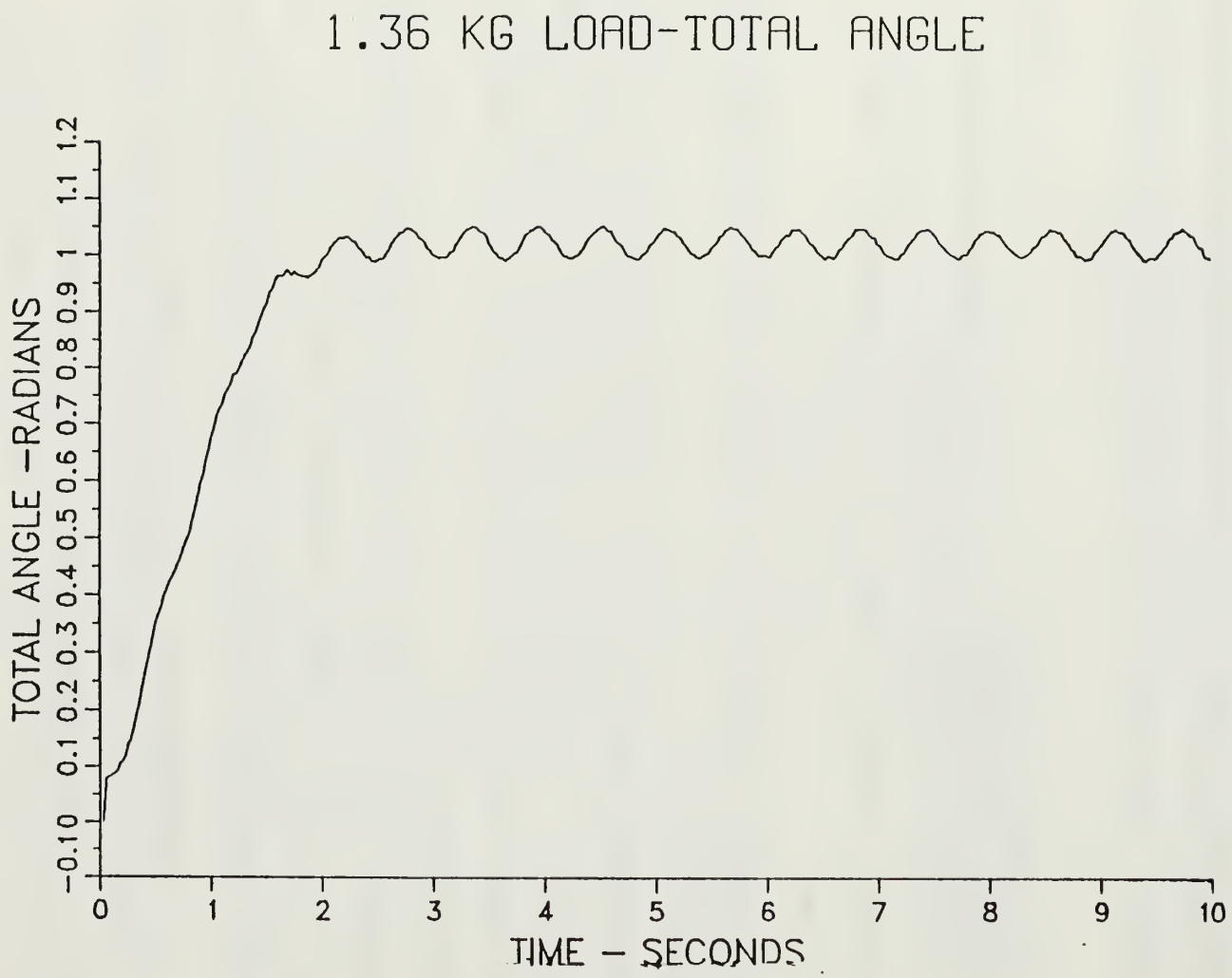

Figure 4.8 - 1.36 Kg At Sampling Rate Of 50 Hz

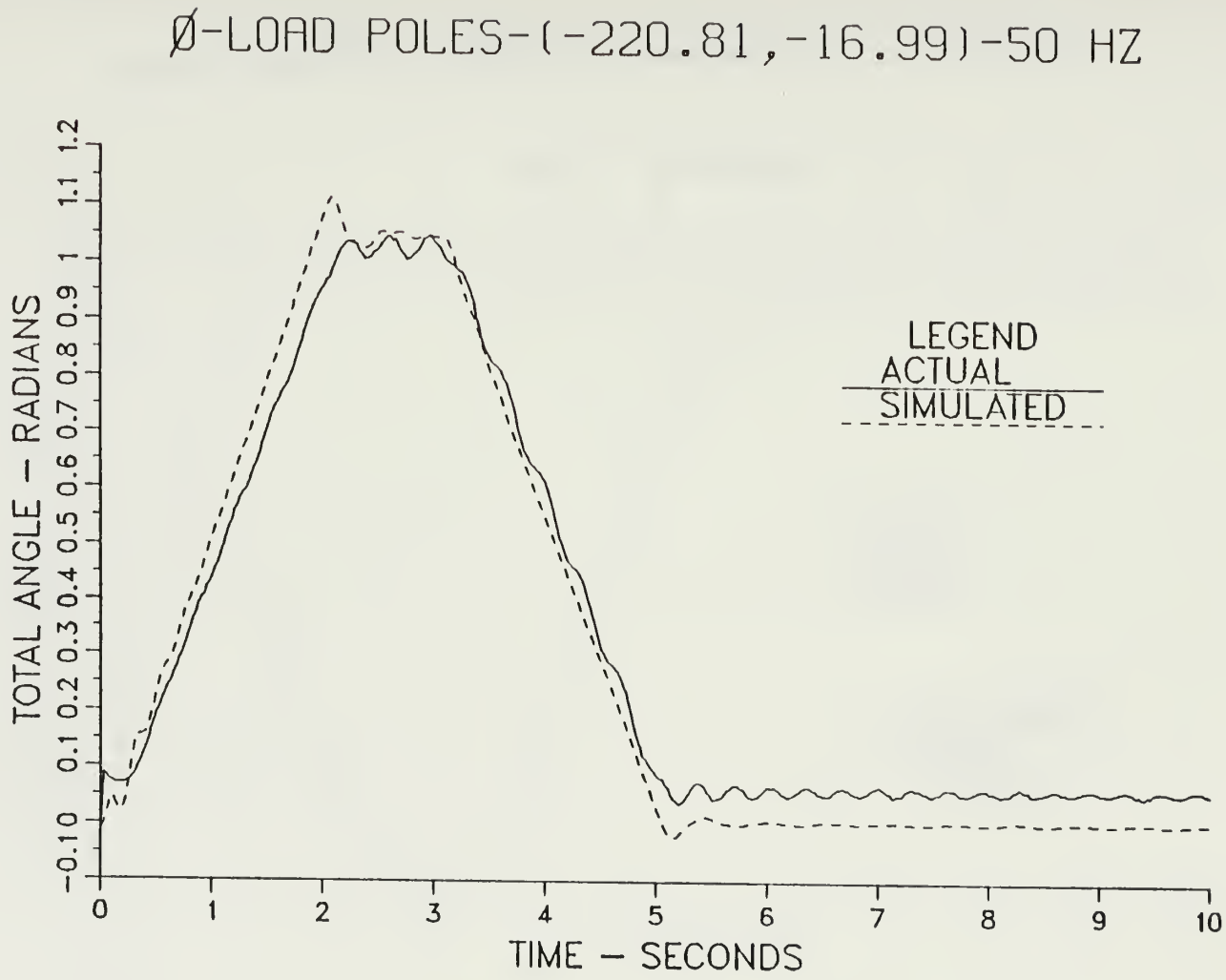

Figure 4.9 - Ø Kg Load Trajectory At 50 Hz

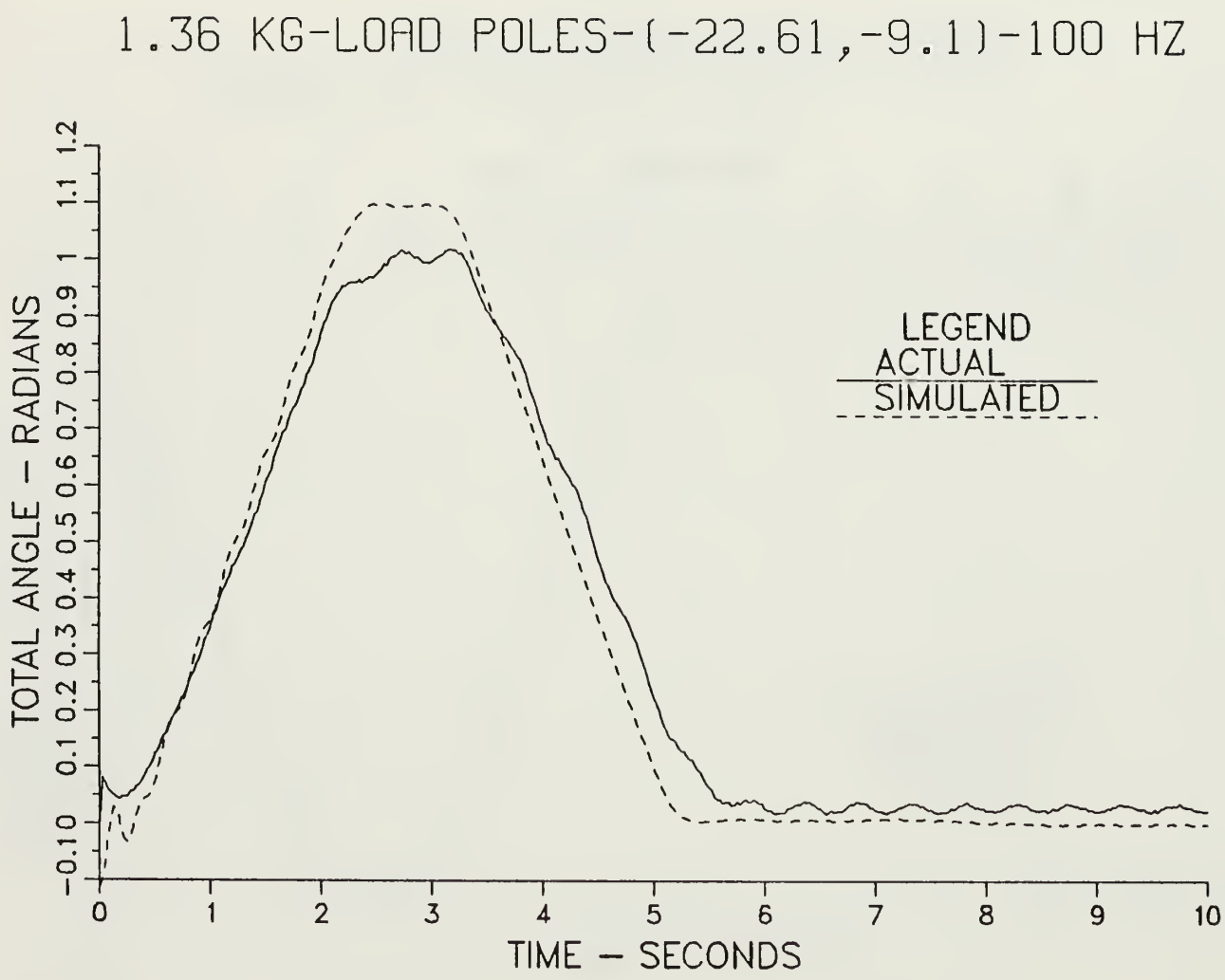

Figure 4.10- 1.36 Kg Load Trajectory Control At 100 Hz

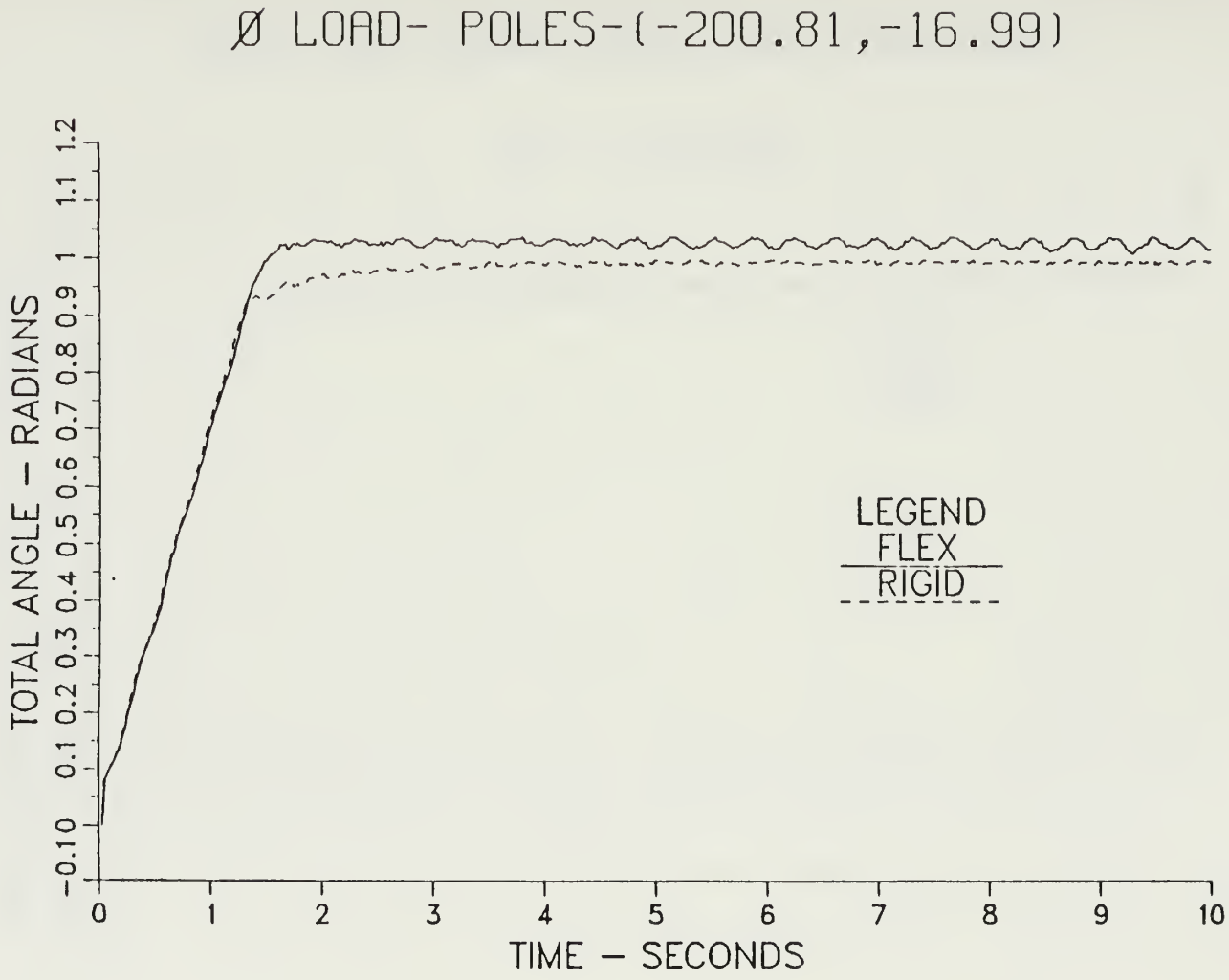

Figure 4.11- Ø Load Flexible-Body Model Vs Rigid-Body Model

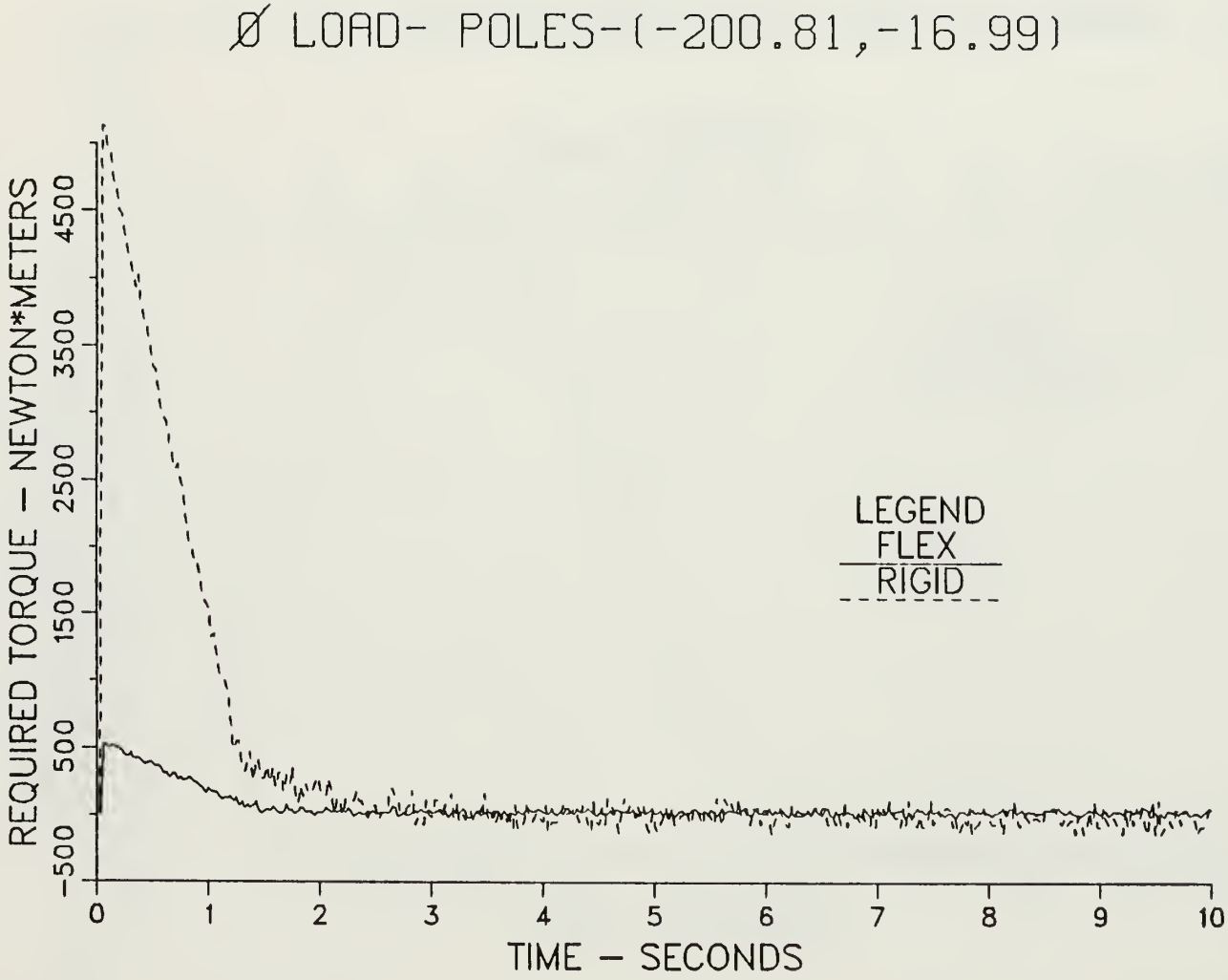

Figure 4.12- Ø Load Flexible-Body Vs Rigid-Body

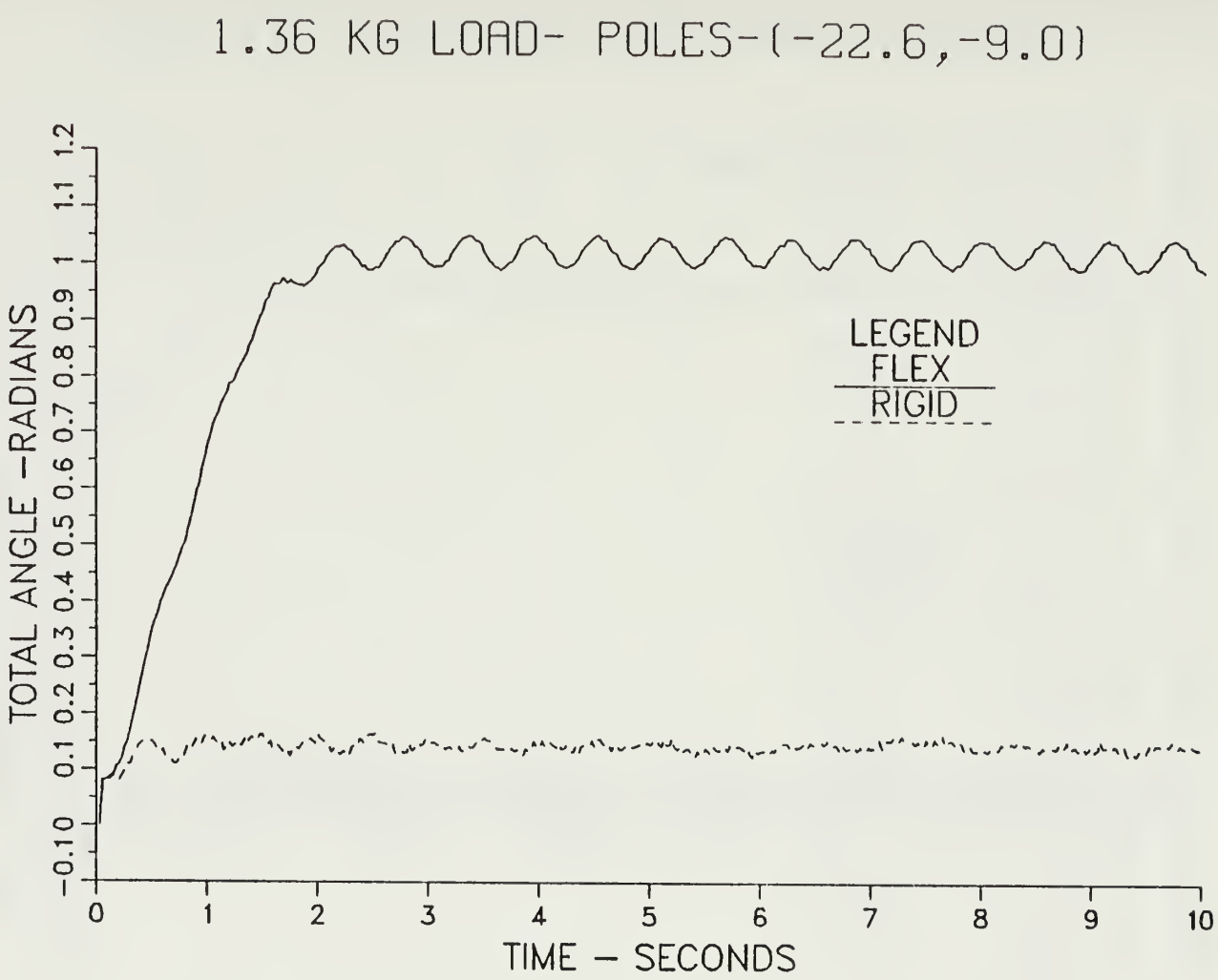

Figure 4.13- 1.36 Kg Load Flexible-Body Vs Rigid-Body

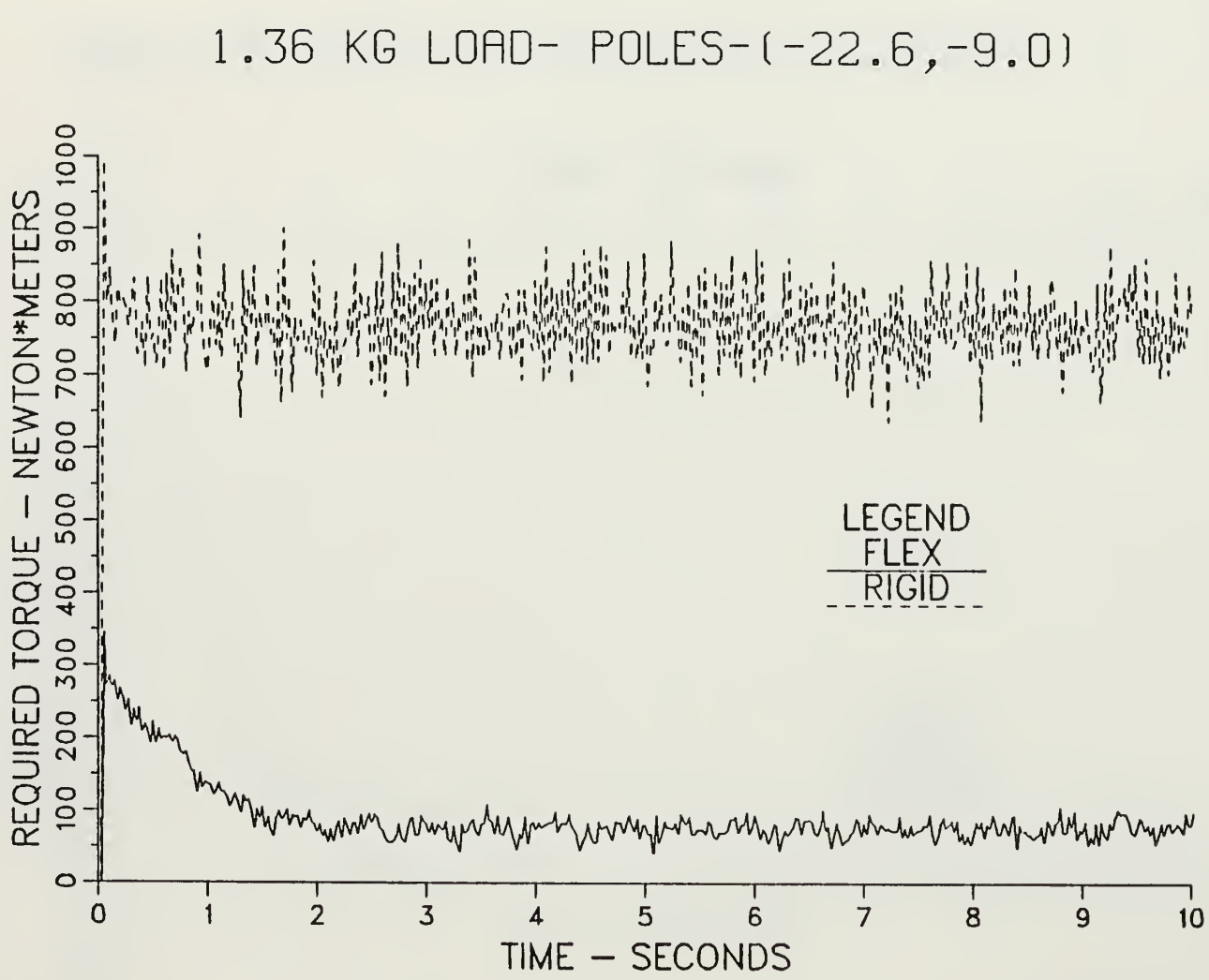

Figure 4.14- 1.36 Kg Load Flexible-Body Vs Rigid-Body

F. RECOMMENDATIONS

It is recommended that;

1. The effects of modal coupling be studied to build a more robust controller.

2. Due to the system's non-linearities, the effects of imposing saturation limits on the arm should be studied.

3. The state space model be expanded to four states with four feedback controls. two states representing the large angle position and velocity and two states representing the small angle position and velocity. This will allow direct control over the small motion component.

4. Extend the present study of the flexible manipulators to include a double-link flexible arm and conduct a more thorough comparison between a flexible-body-model controller and a rigid-body-model controller.

APPENDIX A

SAMPLE PROGRAM FOR SIMULATING THE SYSTEM

APPENDIX A

```
TITLE NO LOAD
CONST       GKV=2.0   ,GKP=1.00,DEA=1.0,RK=867.24828,TAU=0.020
* THESIS COPY
*
*        SIMULATION OF SINGLE LINK FLEXIBLE MANIPULATOR DYNAMICS
*        -------------------------------------------------------
*
*
*    THIS PROGRAM SOLVES THE ERLS FLEXIBLE MANIPULATOR DYNAMICS FOR A
*    SINGLE LINK EXPERIMENTAL ARM. THE EXPERIMENTAL ARM PARAMETERS ARE
*    INPUTTED AND THE HYDRAULIC ACTUATION DYNAMICS ARE INCLUDED IN THE
*    SIMULATION. THE INPUT IS THE CURRENT TO THE SERVOVALVE MOUNTED ON
*    THE HYDRAULIC ACTUATOR AND THE OUTPUT IS THE POSITION OF THE ARM
*    TIP IN THE GLOBAL REFERENCE SYSTEM. THE CODING CONSISTS OF A MAIN
*    PROGRAM AND FIFTEEN SUBROUTINES AND ARE DESCRIBED BELOW.
*
*    THE FOLLOWING PARAMETERS ARE DEFINED:
*    1.A-EFFECTIVE CROSS-SECTIONAL AREA OF FLEXIBLE ARM
*    2.ARRDD-3X3 SECOND TIME DERIVATIVE OF ROTOR RESIDUAL ACCELERATION
*       MATRIX
*    3.ARTH-3X3 ROTOR TRANSFORMATION MATRIX DIFFERENTIATED WITH RESPECT
*      TO THETA
*    4.BE-EFFECTIVE BULK MODULUS OF FLUID
*    5.BIGF-3X1 RIGHT-HAND SIDE VECTOR FOR LARGE AND SMALL MOTION
*       ACCELERATIONS
*    6.BIGM-3X3 MATRIX OF LARGE AND SMALL MOTION ACCELERATION
*       COEFFICIENTS
*    7.CTM-TOTAL LEAKAGE COEFFICIENT OF THE ACTUATOR
*    8.DEFM-DISPLACEMENT DEFORMATION VARIABLE
*    9.DEFMD-TIME DERIVATIVE OF DISPLACEMENT DEFORMATION VARIABLE
*    10.DIFF,QERR,QERR1,FACTOR-DUMMY VARIABLES
*    11.DL1-3X3 DEFORMATION MATRIX
*    12.DL11-3X3 DEFORMATION MATRIX DIFFERENTIATED WITH RESPECT TO THE
*       DISPLACEMENT DEFORMATION VARIABLE
*    13.DL12-3X3 DEFORMATION MATRIX DIFFERENTIATED WITH RESPECT TO THE
*       SLOPE DEFORMATION VARIABLE
*    14.DL1D-3X3 FIRST TIME DERIVATIVE OF DEFORMATION MATRIX
*    15.DM-ACTUATOR DISPLACEMENT
*    16.E-MODULUS OF ELASTICITY OF STEEL
*    17.FN-2X1 RIGHT-HAND SIDE VECTOR FOR SMALL MOTION ACCELERATIONS
*    18.FQ-RIGHT-HAND SIDE FOR LARGE MOTION ACCELERATIONS
*    19.G-3X1 GRAVITATIONAL ACCELERATION VECTOR
*    20.GPOS-3X1 GLOBAL POSITION VECTOR FOR ARM TIP
*    21.H11-1X3 LINK FIRST MOMENT OF INERTIA VECTOR
*    22.H21-2X3 LINK SHAPE MATRIX FIRST MOMENT OF INERTIAVECTOR
*    23.H41-1X3 LOAD FIRST MOMENT OF INERTIA VECTOR
*    24.KCE-TOTAL FLOW PRESSURE COEFFICIENT
*    25.PL-LOAD HYDRAULIC PRESSURE DROP
*    26.PS-HYDRAULIC SUPPLY PRESSURE
*    27.QL-FLOW DELIVERED FROM THE SERVOVALVE
*    28.SLOP-SLOPE DEFORMATION VARIABLE
```

```
*     29. SLOPD-TIME DERIVATIVE OF SLOPE DEFORMATION VARIABLE
*     30. SOL-3X1 VECTOR OF LARGE AND SMALL MOTION ACCELERATIONS
*     31. TE-TORQUE EFFICIENCY
*     32. TH-LARGE MOTION POSITION VARIABLE
*     33. THD-TIME DERIVATIVE OF LARGE MOTION VARIABLE
*     34. TORQUE-APPLIED TORQUE BY ACTUATOR
*     35. U-2X1 ARM TIP DEFORMATION VECTOR INCLUDING DISPLACEMENT AND SLOPE
*     36. UD-2X1 ARM TIP DEFORMATION VECTOR DIFFERENTIATED WITH RESPECT TO
*       TIME
*     37. VT-TOTAL COMPRESSED VOLUME INCLUDING ACTUATOR LINES AND CHAMBERS
*     38. W-3X3 LINK TRANSFORMATION MATRIX
*     39. WD-3X3 FIRST TIME DERIVATIVE OF LINK TRANSFORMATION MATRIX
*     40. WRDD-3X3 SECOND TIME DERIVATIVE OF LINK RESIDUAL ACCELERATION
*       MATRIX
*     41. WTH-3X3 TRANSFORMATION MATRIX DIFFERENTIATED WITH RESPECT TO
*       THETA
*     42. XIFRAC-VARIABLE FRACTIONAL AMOUNT OF INPUT CURRENT TO SERVO-
*       VALVE
*     43. XIINP-CURRENT INPUT EQUAL TO INITIAL AND FRACTIONAL AMOUNTS
*     44. XIL-3X3 INERTIA MATRIX OF THE LOAD
*     45. XIO-INITIAL INPUT CURRENT TO SERVOVALVE
*     46. XIR-3X3 ROTOR INERTIA MATRIX
*     47. XISTEP-STEP INPUT OF FRACTIONAL AMOUNT OF INPUT CURRENT
*     48. XK11-2X2 PARTIAL LINK STIFFNESS MATRIX
*     49. XKN-2X2 LINK STIFFNESS MATRIX
*     50. XKV-SERVOVALVE SIZING CONSTANT
*     51. XLL-LENGTH OF FLEXIBLE ARM
*     52. XML-MASS OF LOAD
*     53. XMNN-2X2 COEFFICIENT MATRIX OF SMALL MOTION ACCELERATIONS IN THE
*       SMALL MOTION DYNAMIC EQUATIONS
*     54. XMNQ-2X1 COEFFICIENT VECTOR OF LARGE MOTION ACCELERATIONS IN THE
*       SMALL MOTION DYNAMIC EQUATIONS
*     55. XMQN-1X2 COEFFICIENT VECTOR OF SMALL MOTION ACCELERATIONS IN THE
*       LARGE MOTION DYNAMICS EQUATION
*     56. XMQQ-COEFFICIENT OF LARGE MOTION ACCELERATION IN THE LARGE MOTION
*       DYNAMICS EQUATION
*     57. XMQQP-2X2 DUMMY MATRIX FOR USE IN FORMULATING THE EQUATIONS OF
*       MOTION
*     58. XMR-MASS OF ACTUATOR ROTOR
*     59. XMU-MASS DENSITY OF STEEL FLEXIBLE ARM
*     60. XMX-FIRST MOMENT OF LOAD WITH RESPECT TO THE LOCAL COORDINATE
*       Y AXIS
*     61. XXI-VARIABLE REPRESENTING INERTIA-LIKE LOAD PROPERTY
*     62. YYI-VARIABLE REPRESENTING INERTIA-LIKE LOAD PROPERTY
*     63. ZI-AREA MOMENT OF INERTIA OF FLEXIBLE ARM
*
*         INITIAL VALUES OF PARAMETERS ARE INPUTTED VIA XINIT SUBROUTINE
*
INITIAL
D      DIMENSION U(1  ),XMQQ(1),XMQQP(1  ),DL1(3,3),WTH(3,3),ARTH(3,3),
D     #XIR(3,3),XMQN(1  ),UD(1  ),H11(1,3),G(3,1),H21(1,3),
D     #WRDD(3,3),DL1D(3,3),WD(3,3),ARRDD(3,3),H41(1,3),XK11(1  ),
D     #          XMNQ(1  ),W(3,3),XMNN(1  ),XKN(1  ),FN(1  ),BIGM(2,2),
D     #BIGF(2,1),XIL(3,3),DL11(3,3),DEFMD(1),SOL(2),THD(1),
D     #         A(1),E(1),ZI(1)              ,FQ(1),GPOS(3),XITH(1),
```

```
D     #XMU(1),XLL(1),XML(1),XMR(1),XMX(1),TH(1),TORQUE(1),DEFM(1),
D     #PS(1),                  CTM(1),VT(1),BE(1),DM(1),XKV(1),TE(1),
D     #QL(1),PL(1),DIFF(1),XIINP(1),QERR1(1),QERR(1),FACTOR(1),XISTEP(1),
D     #  DEIC(1), DETO(1),FTT(1),CTB(1),FCO(1)    ,DEPL(1),
D     # DI(1),DEQ(1),FC1(1),UC(1)
FIXED I
NOSORT
D      COMMON/FCDATA/C1,C2,   A1P,A2P,BETA1,BETA2
*
       C1=0.515462194
       C2=-0.205906794
       A1P=1.362220557
       A2P=0.981867539
       BETA1=1.877920950
       BETA2=4.701142847
        DELS=TAU
*
EXCLUDE U,XMQQ,XMQQP,DL1,WTH,ARTH, ...
       XIR,XMQN,UD,H11,G,H21, ...
       WRDD,DL1D,WD,ARRDD,H41,XK11,  ...
            XMNQ,W,XMNN,XKN,FN,BIGM,  ...
       BIGF,XIL,DL11,DEFMD,SOL,THD,        ...
             A,E,ZI,         FQ,GPOS,XITH,  ...
       XMU,XLL,XML,XMR,XMX,TH,TORQUE,DEFM,  ...
       PS,            CTM,VT,BE,DM,XKV,TE,  ...
       QL,PL,DIFF,XIINP,QERR1,QERR,FACTOR,XISTEP, ...
       DETO,FTT,CTB,DEIC,    DEPL,DEQ,DI,DEFP,DEIC1,DEIC2,...
       C2,A2P,BETA2, C1,A1P,BETA1,FC1
*      INITIALIZATION SUBROUTINE
*
       CALL XINIT(TH,THD,DEFM,DEFMD           ,VO,POSO,A,XML,XMU,...
       XLL,XMR,E,ZI,PS           ,CTM,VT,BE,DM,XKV,TE,QL,...
       PL,PLIC, DEIC)
*
*      WRITE(6,1)XML
*1     FORMAT(G13.5)
DERIVATIVE
*
*         COEFFICIENTS FOR BOTH LARGE AND SMALL MOTION ACCELERATIONS
*         AND THE RIGHT-HAND SIDES ARE COMPUTED IN THE FOLLOWING
*         SUBROUTINES. ALSO,THE HYDRAULIC DYNAMICS ARE INCLUDED
*         IN THE MAIN PROGRAM.
*
NOSORT
*
*         HYDRAULIC DYNAMICS

*      XISTEP(1)=XIFRAC(1)*STEP(0.0)
*      XIINP(1)=XIO(1)+XISTEP(1)
        XIINP(1)=DEIC(1)
       IF(PL(1).GT.PS(1)) GO TO 2
       GO TO 3
 2     PL(1)=PS(1)
 3     QERR1(1)=(XIINP(1)*XKV(1)*DSQRT(PS(1)-PL(1)))-(DM(1)*THD(1))
       QERR(1)=QERR1(1)/CTM(1)
```

```
      DIFF(1)=QERR(1)-PL(1)
      FACTOR(1)=VT(1)/(4.0D0*BE(1)*CTM(1))
      DIFF1(1)=DIFF(1)
SORT
      PL1=INTGRL(PLIC,DIFF1,1)
NOSORT
      PL(1)=PL1(1)/FACTOR(1)
      TORQUE(1)=TE(1)*PL(1)*DM(1)
*     TORQUE(1)=DETO(1)
*      TORQU=TORQUE(1)
*     MATRIX AND VECTOR FORMULATION SUBROUTINE
*
      CALL FORM(W,WTH,WD,DL1,DL1D,XIL,XIR,ARTH,WRDD,ARRDD,U,UD,...
      XMQQP,G,H11,H21,DL11,     H41,XK11,A,XMU,XML,XLL,TH,THD,...
      DEFM,DEFMD            ,E,ZI,XMR,XMX        )
         TP1=TORQUE(1)
*
*     COEFFICIENT OF LARGE MOTION ACCELERATION IN LARGE MOTION DYNAMICS
*     EQUATION SUBROUTINE
*
      CALL XLMMQQ(XMQQ,U,XMQQP,DL1,WTH,ARTH,XIL,XIR,A,XMU,TH,SP,...
      DEFM,MQQ1,TP)
         T6=TP
*
*     COEFFICIENTS OF SMALL MOTION ACCELERATIONS IN LARGE MOTION DYNAMICS
*     EQUATION SUBROUTINE
*
      CALL XLMMQN(XMQN,A,XMU,XML,XLL,XMX,     DEFM        )
*
*     RIGHT-HAND SIDE FOR LARGE MOTION DYNAMICS EQUATION SUBROUTINE
*
      CALL XLMFQ(FQ,U,XMQQP,DL1,WTH,ARTH,XIL,XIR,UD,H11,G,H21,WRDD,...
      DL1D,WD,ARRDD,H41,TH,THD,DEFM,DEFMD,          A,XMU,XML,XLL,...
      TORQUE,FTT)
*
*     LINK STIFFNESS MATRIX SUBROUTINE
*
      CALL SMKN(XKN,XK11,XMQQP,A,XMU,THD)
*
*
*
*     COEFFICIENTS OF LARGE MOTION ACCELERATION IN SMALL MOTION
*     DYNAMICS EQUATIONS SUBROUTINE
*
      CALL SMMNQ(XMNQ,DL1,WTH,XIL,DL11,     W,TH,DEFM     ,A,XMU,...
      XLL)
         MNQ=XMNQ(1)
*
*     RIGHT-HAND SIDE OF SMALL MOTION DYNAMICS EQUATIONS SUBROUTINE
*
      CALL SMFN(FN,H21,W,G,WRDD,DL1,XIL,DL11     ,WD,DL1D,H41,TH,...
      THD,DEFM,DEFMD          )
*
*     COEFFICIENTS OF SMALL MOTION ACCELERATIONS IN SMALL MOTION DYNAMICS
```

```
*     EQUATIONS SUBROUTINE
*
      CALL SMMNN(XMNN,XMQQP,XML,A,XMU            )
         MNN=XMNN(1)
*
*     ACCELERATION COEFFICIENTS MATRIX AND RIGHT-HAND SIDE VECTOR
*     FORMULATION SUBROUTINE
*
      CALL BIGFOR(BIGM,BIGF,XMQQ,XMQN,FQ,XMNQ,XMNN,XKN,FN,U,DEFMD)
*
*     LINEAR EQUATION SOLVER FOR ACCELERATIONS SUBROUTINE
*
      CALL XLEQ(BIGM,BIGF,SOL)
*
*     TRANSFORMATION FROM LOCAL COORDINATE TO GLOBAL COORDINATE TIP
*     POSITION SUBROUTINE
*
*     CALL GLOB(GPOS,W,DEFM)


*     INTEGRATE ACCELERATIONS AND THEN VELOCITY TO GET LARGE MOTION
*     ANGULAR POSITION AND SMALL MOTION,LOCAL COORDINATE,TIP POSITION
*
      DO 5 I=1,2
      SOL1(I)=SOL(I)
   5  CONTINUE
SORT
      VEL=INTGRL(VO,SOL1,2)
NOSORT
      THD(1)=VEL(1)
      DEFMD(1)=VEL(2)
*     SLOPD(1)=VEL(3)
      DO 10 I=1,2
      VEL1(I)=VEL(I)
  10  CONTINUE
SORT
      POS=INTGRL(POSO,VEL1,2)
NOSORT
      TH(1)=POS(1)
      DEFM(1)=POS(2)
      AC1=SOL1(1)
      AC2=SOL1(2)
      VE1=VEL(1)
      VE2=VEL(2)
      PO1=POS(1)
      PO2=POS(2)
      LA=PO1
      S=PO2
*
*     CALL  TRAIN(POS,XLL,THICK,STRANE)

*     STRAIN=STRANE(1)
*
           TAG2=AC1+AC2/0.9985D0
           TAG1=VE1+VE2/0.9985D0
```

```
            TAG =PO1  +PO2/0.9985D0
*
*
*************************************************************************
*         CALL CCO(XMQQ,XMQN,XMNN,XMNQ,FTT,FN,XKN,...
*              DEFM,CTB,FCO,FC1,DEFMD)
          T5=CTB(1)
*
SAMPLE
*            DETO(1)=RK+3.9595*PO1-51.2915*VE1
*            DETO(1)=1.66898*(-70.00*VE1+1225*(1-PO1))+23.7854*SIN(PO1)
             UC(1)=RK-845.7607*TAG-60.0000*TAG1
             DETO(1)=UC(1) + 28.4392
*          CALL DESCU(DETO,CTB,TAG1,TAG,FCO,GKV,GKP,DEA)
*
*************************************************************************
*          INVERSE HYDRAUIC  DYNAMIC
*
           DEPL(1)=DETO(1)/(TE(1)*DM(1))
             DI(1)=VT(1)*(DEPL(1)-  PL(1))/(4.0D0*BE(1)*0.006D0)
             DEQ(1)=DM(1)*THD(1) +CTM(1)*DEPL(1) +DI(1)
             DEFP =  (PS(1)- DEPL(1))
             IF(DEFP .LT.0.0)  THEN
               DEIC(1)=4.0D0
               GOTO 15
               ENDIF
             DEIC1=DSQRT(DEFP)
*             GO TO 14
*3            DEFP=-DEFP
*             DEIC1=DSQRT(DEFP)
            DEIC2=DEIC1*XKV(1)
            DEIC(1)=DEQ(1)/(DEIC2     )
             TORK=DETO(1)
             IF(DEIC(1) .GT.  4.0D0) DEIC(1)=4.0D0
             IF(DEIC(1) .LT.  -4.0D0) DEIC(1)=-4.0D0
15           CU=DEIC(1)
TERMINAL
METHOD ADAMS
CONTROL FINTIM=5.0000 ,DELT=0.005
PRINT 25.0E-3,TAG,PO1,PO2,TORK,DEIC
SAVE  25.0E-3,TAG,PO1,PO2,TORK,DEIC
*ND
*ARAM GKV=4.00  , GKP=4.00
*ND
*ARAM GKV=8.00 ,  GKP=16.00
*RAPH (G1,DE=TEK618) TIME(UN=SEC) ,LA(LO=-0.12,SC=0.15),...
*                                  S(LO=-0.12,SC=0.15)
*ABEL (G1) LOAD=0.0000 KG
*RAPH (G2,DE=TEK618) TIME(UN=SEC),TAG,TAG1
*ABEL (G2) LOAD=0.0000 KG
*RAPH (G3,DE=TEK618)TIME,DEA
*ABEL (G3) LOAD=0.0000 KG
*RAPH (G4,DE=TEK618)TIME(UN=SEC),TAG(UN=RAD)
*ABEL (G4) LOAD=0.0000 KG
*RAPH (G5,DE=TEK618) TIME(UN=SEC),CU(UN=MA),TORK(UN=N-M)
```

```
*ABEL (G5) LOAD=0.0000KG
*RAPH (G1,DE=TEK618) TAG1,T5
*ABEL (G1) LOAD=0.0000 KG
END
STOP
*
*    LISTING OF SUBROUTINES
*
FORTRAN
*
*.....................................................................
*
      SUBROUTINE XINIT(TH,THD,DEFM,DEFMD   ,VO,POSO,A,ML,MU,LL,MR,
     #E,ZI,PS,          CTM,VT,BE,DM,KV,TE,QL,PL,PLIC,DEIC)
      REAL*8 VO(2),POSO(2),ML,MU,LL,MR,TH,THD,DEFM,DEFMD,
     #A,E,ZI,ITORQ,PS,          CTM,VT,BE,DM,KV,TE,QL,PL,PLIC,
     #C1,          AP1,    BETA1                          ,DEIC  ,
     #DEPL  , DEQ,DI,DEFP,DEIC1,DEIC2
      ITORQ= 23.893030030000000
      DM=6.2271D-05
      TE=.90000000000000
      PL=ITORQ/(DM*TE)
      PLIC=PL
      CTM=3.7064772D-13
      QL=CTM*PL
      KV=2.402963D-09
      PS=1.37888D+07
*     IO=QL/(KV*DSQRT(PS-PL))
*     IMAX=10.000000000000000
*     IFRAC=.4D0*(IMAX-IO)
      VT=3.05127D-04
      BE=690.D6
      A=6.17795D-04
      ML=0.0000000000000
      MU=7861.05000000000000
      LL=0.99850000000000
*     STRANE(1)=0.00000000000000
      MR=9.00011451000000
      E=2.0D11
      ZI=4.065D-10
      VO(1)=0.000000000000000
      VO(2)=0.000000000000000
      POSO(1)=.050931116
      POSO(2)=-.0610213450000
      TH=POSO(1)
      THD=VO(1)
      DEFM=POSO(2)
      DEFMD=VO(2)
       DEIC=0.2
*     DETO= 40.5
*     PEPL=PLIC
      RETURN
      END
*
*.....................................................................
```

```
*
*      DOUBLE PRECISION FUNCTION ONE(X)
*      REAL*8 C1,A1P,BETA1
*      COMMON/FCDATA/C1,A1P,BETA1
*
*      ONE=                                    C1*(A1P*(COS(BETA1*X)+COSH
*     #                     (BETA1*X))+(SIN(BETA1*X)+SINH(BETA1*X)))
*      RETURN
*      END

*      DOUBLE PRECISION FUNCTION TWO(X)
*      REAL*8 C1,A1P,BETA1
*      COMMON/FCDATA/C1,A1P,BETA1
*
*      TWO=                                  (C1*(A1P*(COS(BETA1*X)+COSH
*     #                     (BETA1*X))+(SIN(BETA1*X)+SINH(BETA1*X))))
*     #                        **2
*      RETURN
*      END
*
*
*
*      DOUBLE PRECISION FUNCTION SIX(X)
*      REAL*8 C1,A1P,BETA1
*      COMMON/FCDATA/C1,A1P,BETA1
*
*      SIX=                .9985D0*(C1*(A1P*(COS(BETA1*X)+COSH
*     #                     (BETA1*X))+(SIN(BETA1*X)+SINH(BETA1*X))))+
*     #                                X*(C1*(A1P*(COS(BETA1*X)+COSH
*     #                     (BETA1*X))+(SIN(BETA1*X)+SINH(BETA1*X))))
*      RETURN
*      END
*
*      DOUBLE PRECISION FUNCTION EIGHT(X)
*      REAL*8 C1,A1P,BETA1
*      COMMON/FCDATA/C1,A1P,BETA1
*
*      EIGHT=                                (C1*BETA1*BETA1*(A1P*(-COS
*     #          (BETA1*X)+COSH(BETA1*X))+(-SIN(BETA1*X)+SINH(BETA1*X))))
*     #               **2
*      RETURN
*      END
*
       DOUBLE PRECISION FUNCTION ONE(X)
       REAL*8 C1,C2,      A1P,A2P,BETA1,BETA2
       COMMON/FCDATA/C1,C2       ,A1P,A2P,BETA1,BETA2

       ONE=                                  C1*(A1P*(COS(BETA1*X)+COSH
      #                     (BETA1*X))+(SIN(BETA1*X)+SINH(BETA1*X)))+
      #                        C2*(A2P*(COS(BETA2*X)+COSH(BETA2*X))+
      #                                (SIN(BETA2*X)+SINH(BETA2*X)))
       RETURN
       END
C
       DOUBLE PRECISION FUNCTION TWO(X)
```

```
      REAL*8 C1,C2,      A1P,A2P,BETA1,BETA2
      COMMON/FCDATA/C1,C2,      A1P,A2P,BETA1,BETA2

      TWO=                                (C1*(A1P*(COS(BETA1*X)+COSH
     #                    (BETA1*X))+(SIN(BETA1*X)+SINH(BETA1*X)))+
     #                       C2*(A2P*(COS(BETA2*X)+COSH(BETA2*X))+
     #                               (SIN(BETA2*X)+SINH(BETA2*X))))**2
      RETURN
      END
      DOUBLE PRECISION FUNCTION SIX(X)
      REAL*8 C1,C2,      A1P,A2P,BETA1,BETA2
      COMMON/FCDATA/C1,C2,      A1P,A2P,BETA1,BETA2

      SIX=                         .9985*(C1*(A1P*(COS(BETA1*X)+COSH
     #                    (BETA1*X))+(SIN(BETA1*X)+SINH(BETA1*X)))+
     #                       C2*(A2P*(COS(BETA2*X)+COSH(BETA2*X))+
     #                               (SIN(BETA2*X)+SINH(BETA2*X))))+
     #                                X*(C1*(A1P*(COS(BETA1*X)+COSH
     #                    (BETA1*X))+(SIN(BETA1*X)+SINH(BETA1*X)))+
     #                       C2*(A2P*(COS(BETA2*X)+COSH(BETA2*X))+
     #                               (SIN(BETA2*X)+SINH(BETA2*X))))
      RETURN
      END


      DOUBLE PRECISION FUNCTION EIGHT(X)
      REAL*8 C1,C2,      A1P,A2P,BETA1,BETA2
      COMMON/FCDATA/C1,C2,      A1P,A2P,BETA1,BETA2

      EIGHT=                              (C1*BETA1*BETA1*(A1P*(-COS
     #         (BETA1*X)+COSH(BETA1*X))+(-SIN(BETA1*X)+SINH(BETA1*X)))+
     #              C2*BETA2*BETA2*(A2P*(-COS(BETA2*X)+COSH(BETA2*X))+
     #                              (-SIN(BETA2*X)+SINH(BETA2*X))))**2
      RETURN
      END
*.....................................................................
*
      SUBROUTINE FORM(W,WTH,WD,DL1,DL1D,XIL,XIR,ARTH,WRDD,ARRDD,U,UD,
     #XMQQP,G,H11,H21,DL11      ,H41,XK11,A,MU,ML,LL,TH,THD,DEFM,DEFMD,
     #          E,ZI,MR,MX       )
      REAL*8 W(3,3),WTH(3,3),WD(3,3),DL1(3,3),DL1D(3,3),XIL(3,3)
      REAL*8 XIR(3,3),ARTH(3,3),WRDD(3,3),ARRDD(3,3),U      ,UD
      REAL*8 XMQQP      ,G(3,1),H11(1,3),H21(1,3),DL11(3,3)
      REAL*8 H41(1,3),XK11      ,MU,ML,LL,MR,MX,TH
      REAL*8 THD,DEFM,DEFMD                      ,A,E,ZI
      REAL*8 C1,A1P,BETA1
      REAL*8 ONE,TWO,EIGHT,I1,I4,I6
      EXTERNAL ONE,TWO,EIGHT
      COMMON/FCDATA/C1,A1P,BETA1
      W(1,1)=1.00000000000000
      W(1,2)=0.00000000000000
      W(1,3)=0.00000000000000
      W(2,1)=LL*DCOS(TH)
      W(2,2)=DCOS(TH)
      W(2,3)=-DSIN(TH)
```

```
      W(3,1)=LL*DSIN(TH)
      W(3,2)=DSIN(TH)
      W(3,3)=DCOS(TH)
      WTH(1,1)=0.00000000000000
      WTH(1,2)=0.00000000000000
      WTH(1,3)=0.00000000000000
      WTH(2,1)=-LL*DSIN(TH)
      WTH(2,2)=-DSIN(TH)
      WTH(2,3)=-DCOS(TH)
      WTH(3,1)=LL*DCOS(TH)
      WTH(3,2)=DCOS(TH)
      WTH(3,3)=-DSIN(TH)
      WD(1,1)=0.00000000000000
      WD(1,2)=0.00000000000000
      WD(1,3)=0.00000000000000
      WD(2,1)=-LL*DSIN(TH)*THD
      WD(2,2)=-DSIN(TH)*THD
      WD(2,3)=-DCOS(TH)*THD
      WD(3,1)=LL*DCOS(TH)*THD
      WD(3,2)=DCOS(TH)*THD
      WD(3,3)=-DSIN(TH)*THD
      DL1(1,1)=1.00000000000000
      DL1(1,2)=0.00000000000000
      DL1(1,3)=0.00000000000000
      DL1(2,1)=0.00000000000000
      DL1(2,2)=1.00000000000000
*     DL1(2,3)=-1.378573350D0*DEFM
      DL1(2,3)=-0.00000*DEFM
      DL1(3,1)=DEFM
*     DL1(3,2)= 1.378573350D0*DEFM
      DL1(3,2)=0.00000*DEFM
      DL1(3,3)=1.00000000000000
      DL1D(1,1)=0.00000000000000
      DL1D(1,2)=0.00000000000000
      DL1D(1,3)=0.00000000000000
      DL1D(2,1)=0.00000000000000
      DL1D(2,2)=0.00000
*     DL1D(2,3)=-1.378573350D0*DEFMD
      DL1D(2,3)=-0.0000000*DEFMD
      DL1D(3,1)=DEFMD
*     DL1D(3,2)= 1.378573350D0*DEFMD
      DL1D(3,2)=0.0000000*DEFMD
      DL1D(3,3)=0.00000000000000
      XIL(1,1)=ML
      XIL(1,2)=0.00000000000000
      XIL(1,3)=.00000000000000
      XIL(2,1)=0.00000000000000
      XIL(2,2)=0.0000000000
      XIL(2,3)=0.00000000000000
      XIL(3,1)=.00000000000000
      XIL(3,2)=0.00000000000000
      XIL(3,3)=0.0000000000
      MX=0.00000000000000
*     XXI=0.0D0
*     YYI=0.0D0
```

```
      XIR(1,1)=MR
      XIR(1,2)=0.000000000000000
      XIR(1,3)=0.000000000000000
      XIR(2,1)=0.000000000000000
      XIR(2,2)=.027467130000000
      XIR(2,3)=0.000000000000000
      XIR(3,1)=0.000000000000000
      XIR(3,2)=0.000000000000000
      XIR(3,3)=.02746713000000
      ARTH(1,1)=0.00000000000000
      ARTH(1,2)=0.00000000000000
      ARTH(1,3)=0.00000000000000
      ARTH(2,1)=0.00000000000000
      ARTH(2,2)=-DSIN(TH)
      ARTH(2,3)=-DCOS(TH)
      ARTH(3,1)=0.00000000000000
      ARTH(3,2)=DCOS(TH)
      ARTH(3,3)=-DSIN(TH)
      WRDD(1,1)=0.00000000000000
      WRDD(1,2)=0.00000000000000
      WRDD(1,3)=0.00000000000000
      WRDD(2,1)=-LL*DCOS(TH)*(THD**2)
      WRDD(2,2)=-DCOS(TH)*(THD**2)
      WRDD(2,3)=DSIN(TH)*(THD**2)
      WRDD(3,1)=-LL*DSIN(TH)*(THD**2)
      WRDD(3,2)=-DSIN(TH)*(THD**2)
      WRDD(3,3)=-DCOS(TH)*(THD**2)
      ARRDD(1,1)=0.00000000000000
      ARRDD(1,2)=0.00000000000000
      ARRDD(1,3)=0.00000000000000
      ARRDD(2,1)=0.00000000000000
      ARRDD(2,2)=-DCOS(TH)*(THD**2)
      ARRDD(2,3)=DSIN(TH)*(THD**2)
      ARRDD(3,1)=0.00000000000000
      ARRDD(3,2)=-DSIN(TH)*(THD**2)
      ARRDD(3,3)=-DCOS(TH)*(THD**2)
      U     =DEFM
      UD     =DEFMD
      CALL DQG4(-LL,0.D0,TWO,I1)
      XMQQP      =I1
*     XMQQP  =0.3714286
      G(1,1)=0.00000000000000
      G(2,1)=0.00000000000000
      G(3,1)=-9.80660000000000
      H11(1,1)=4.85651900000000
      H11(1,2)=-2.42825869300000
      H11(1,3)=0.00000000000000
      H21(1,1)=0.00000000000000
      H21(1,2)=0.00000000000000
      CALL DQG4(-LL,-0.D0,ONE ,I4)
      H21(1,3)=A*MU*I4
*     H21(1,3)=A*MU*(0.50000)
      DO 50 I=1,3
      DO 60 J=1,3
      DL11(I,J)=0.00000000000000
```

```
   60  CONTINUE
   50  CONTINUE
       DL11(3,1)=1.00000000000000
       H41(1,1)=ML
       H41(1,2)=0.000000000000000
       H41(1,3)=.000000000000000000
       CALL DQG4(-LL,0.D0,EIGHT,I6)
       XK11     =I6*E*ZI
       RETURN
       END
*
*.....................................................................
*
       SUBROUTINE XLMMQQ(XMQQ,U,XMQQP,DL1,WTH,ARTH,XIL,XIR,A,MU,TH,SP,DEF
      1M,MQQ1,TP)
       REAL*8 XMQQ,UT      ,P      ,DL1T(3,3),WTHT(3,3),ARTHT(3,3),P1(3,3)
       REAL*8 P2(3,3),P3(3,3),P4(3,3),P5(3,3),P6(3,3),P7(3,3),MU
       REAL*8 U      ,XMQQP      ,DL1(3,3),WTH(3,3),ARTH(3,3),XIL(3,3)
       REAL*8 XIR(3,3),A,TH,DEFM,      SP,TP,MQQ1,MQQ2
       M=2
       L=1
       N=3
       XMQQ=0.00000000000000
       CALL TRANS(U,UT,L,L)
       CALL MATMUL(UT,XMQQP,L,L,L,P)
       CALL MATMUL(P,U,L,L,L,SP)
       CALL TRANS(DL1,DL1T,N,N)
       CALL TRANS(ARTH,ARTHT,N,N)
       CALL TRANS(WTH,WTHT,N,N)
       CALL MATMUL(WTH,DL1,N,N,N,P1)
       CALL MATMUL(P1,XIL,N,N,N,P2)
       CALL MATMUL(P2,DL1T,N,N,N,P3)
       CALL MATMUL(P3,WTHT,N,N,N,P4)
       CALL MATMUL(ARTH,XIR,N,N,N,P5)
       CALL MATMUL(P5,ARTHT,N,N,N,P6)
       CALL MATADD(P4,P6,N,N,P7)
       CALL TRACE(P7,N,TP)
       MQQ1=(1.D0/3.D0)*A*MU
       MQQ2=A*MU*SP
       XMQQ= MQQ1+MQQ2 +  TP

*      WRITE(*,*)SP

       RETURN
       END
*
*.....................................................................
*
       SUBROUTINE XLMMQN(XMQN,A,MU,ML,LL,MX      ,DEFM          )
       REAL*8 XMQN      ,MU,ML,LL,MX,A,      DEFM
       REAL*8 C1,A1P,BETA1,SIX,I9 ,C2,A2P,BETA2
       EXTERNAL SIX
       COMMON/FCDATA/C1,C2,A1P,A2P ,BETA1   ,BETA2
       CALL DQG4(-LL,0.D0,SIX ,I9)
           XMQN =(ML*LL)+MX+(A*MU* I9)
```

```
      RETURN
      END
*
*.....................................................................
*
      SUBROUTINE XLMFQ(FQ,U,XMQQP,DL1,WTH,ARTH,XIL,XIR,UD,H11,G,H21,
     #WRDD,DL1D,WD,ARRDD,H41,TH,THD,DEFM,DEFMD,          A,MU,ML,LL,
     #TORQUE,FTT)
      REAL*8 FQP      ,P      ,P1(1,3),P2(1,3),P3(1,3),P4(3,3),P5(3,3)
      REAL*8 P6(3,3),P7(3,3),P8(3,3),P9(3,3),P10(3,3),P11(1,3),P12(1,3)
      REAL*8 FPFH(3,3),FHP(3,3),FPT(3,3),DL1DT(3,3),WDT(3,3),ARRDDT(3,3)
      REAL*8 UT      ,DL1T(3,3),WRDDT(3,3),FPF(3,3),FPS(3,3),WTHT(3,3)
      REAL*8 U      ,XMQQP      ,DL1(3,3),WTH(3,3),ARTH(3,3),XIL(3,3)
      REAL*8 XIR(3,3),UD      ,H11(1,3),G(3,1),H21(1,3),WRDD(3,3)
      REAL*8 DL1D(3,3),WD(3,3),ARRDD(3,3),H41(1,3),MU,LL,ML
      REAL*8 A,TORQUE,FQ,TH,THD,DEFM,DEFMD
      REAL*8 FP1,SP1,TP1,TFP1,FTHP1,FTT
      M=2
      L=1
      N=3
      CALL TRANS(U,UT,L,L)
      FQP      =XMQQP      *THD
      CALL MATMUL(UT,FQP,L,L,L,P)
      CALL MATMUL(P,UD,L,L,L,FP1)
      CALL TRANS(WTH,WTHT,N,N)
      CALL MATMUL(H11,WTHT,L,N,N,P1)
      CALL MATMUL(P1,G,L,N,L,SP1)
      CALL MATMUL(UT,H21,L,L,N,P2)
      CALL MATMUL(P2,WTHT,L,N,N,P3)
      CALL MATMUL(P3,G,L,N,L,TP1)
      CALL TRANS(DL1,DL1T,N,N)
      CALL TRANS(WRDD,WRDDT,N,N)
      CALL MATMUL(WTH,DL1,N,N,N,P4)
      CALL MATMUL(P4,XIL,N,N,N,P5)
      CALL MATMUL(P5,DL1T,N,N,N,P6)
      CALL MATMUL(P6,WRDDT,N,N,N,FPF)
      CALL TRANS(DL1D,DL1DT,N,N)
      CALL TRANS(WD,WDT,N,N)
      CALL MATMUL(WTH,DL1,N,N,N,P7)
      CALL MATMUL(P7,XIL,N,N,N,P8)
      CALL MATMUL(P8,DL1DT,N,N,N,P9)
      CALL MATMUL(P9,WDT,N,N,N,FPS)
      CALL TRANS(ARRDD,ARRDDT,N,N)
      CALL MATMUL(ARTH,XIR,N,N,N,P10)
      CALL MATMUL(P10,ARRDDT,N,N,N,FPT)
      DO 30 I=1,3
      DO 40 J=1,3
      FPS(I,J)=FPS(I,J)*2.
   40 CONTINUE
   30 CONTINUE
      CALL MATADD(FPF,FPS,N,N,FPFH)
      CALL MATADD(FPFH,FPT,N,N,FHP)
      CALL TRACE(FHP,N,TFP1)
      CALL MATMUL(H41,DL1T,L,N,N,P11)
      CALL MATMUL(P11,WTHT,L,N,N,P12)
```

```
      CALL MATMUL(P12,G,L,N,L,FTHP1)
      FTT=(-2.*A*MU*FP1)+ SP1+ TP1- TFP1+ FTHP1
*     FTT=SP1+TP1-TFP1+FTHP1
      FQ=FTT+TORQUE
      RETURN
      END
*
*.....................................................................
*
      SUBROUTINE SMKN(XKN,XK11,XMQQP,A,MU,THD)
      REAL*8 XKN      ,KNP      ,XMQQP      ,XK11      ,A,THD,MU
      KNP      =XMQQP      *(-A)*MU*(THD**2)
      XKN      =KNP      +XK11
      RETURN
      END
*
*.....................................................................
*
      SUBROUTINE SMMNQ(XMNQ,DL1,WTH,XIL,DL11,      W,TH,DEFM      ,A,MU,
     #LL)
      REAL*8 XMNQ                  ,DL11T(3,3),WT(3,3),P1(3,3),P2(3,3)
      REAL*8 P3(3,3),P4(3,3)                  ,DL1(3,3),WTH(3,3),XIL(3,3)
      REAL*8 W(3,3),DL11(3,3),           TH,DEFM      ,A,MU,LL
      REAL*8 C1,A1P,BETA1,TFP1
      REAL*8 SIX, I11           ,C2,A2P,BETA2
      EXTERNAL SIX
      COMMON/FCDATA/C1,C2, A1P,A2P,BETA1   ,BETA2
      M=2
      L=1
      N=3
      CALL TRANS(DL11,DL11T,N,N)
      CALL TRANS(W,WT,N,N)
      CALL MATMUL(WTH,DL1,N,N,N,P1)
      CALL MATMUL(P1,XIL,N,N,N,P2)
      CALL MATMUL(P2,DL11T,N,N,N,P3)
      CALL MATMUL(P3,WT,N,N,N,P4)
      CALL TRACE(P4,N,TFP1)

*     WRITE(*,*)LL

      CALL DQG4(-LL,0.D0,SIX ,I11)
*     XMNQ      =TFP1 + (I11*A*MU) -(A*MU*0.05*1.378573)
         XMNQ=TFP1+(A*MU*I11)
*     WRITE(*,*)I11

      RETURN
      END
*
*.....................................................................
*
      SUBROUTINE SMFN(FN,H21,W,G,WRDD,DL1,XIL,DL11      ,WD,DL1D,H41,TH,
     #THD,DEFM,DEFMD            )
      REAL*8 FN      ,P1(1,3),P2(3,3),P3(3,3),P4(3,3),P5(3,3),P6(3,3)
      REAL*8 P7(3,3),P8(3,3),P9(3,3)
      REAL*8 P14(1,3),P15(1,3)                  ,TP      ,FP      ,SP
```

```
      REAL*8 FN1(3,3)          ,G(3,1),H21(1,3),WRDD(3,3),DL1D(3,3)
      REAL*8 WD(3,3),H41(1,3),XIL(3,3),W(3,3),DL11(3,3)
      REAL*8 DL1(3,3),DL11T(3,3)            ,WT(3,3)
      REAL*8 TH,THD,DEFM,DEFMD             ,TFN1      ,FN3
      M=2
      L=1
      N=3
      CALL TRANS(W,WT,N,N)
      CALL MATMUL(H21,WT,L,N,N,P1)
      CALL MATMUL(P1,G,L,N,L,FP)
      CALL TRANS(DL11,DL11T,N,N)
      CALL MATMUL(WRDD,DL1,N,N,N,P2)
      CALL MATMUL(P2,XIL,N,N,N,P3)
      CALL MATMUL(P3,DL11T,N,N,N,P4)
      CALL MATMUL(P4,WT,N,N,N,P5)
      CALL MATMUL(WD,DL1D,N,N,N,P6)
      CALL MATMUL(P6,XIL,N,N,N,P7)
      CALL MATMUL(P7,DL11T,N,N,N,P8)
      CALL MATMUL(P8,WT,N,N,N,P9)
      DO 10 I=1,3
      DO 20 J=1,3
      P9(I,J)= P9(I,J)*2.
   20 CONTINUE
   10 CONTINUE
      CALL MATADD(P5,P9,N,N,FN1)
      CALL TRACE(FN1,N,TFN1)
      SP     =TFN1
      CALL MATMUL(H41,DL11T,L,N,N,P14)
      CALL MATMUL(P14,WT,L,N,N,P15)
      CALL MATMUL(P15,G,L,N,L,FN3)
      TP     =FN3
      FN     =FP       - SP       + TP
      RETURN
      END
*
*.....................................................................
*
      SUBROUTINE SMMNN(XMNN,XMQQP,ML,A,MU            )
      REAL*8 XMNN      ,XMQQP      ,ML,MU,A
*     DO 10 I=1,2
*     DO 20 J=1,2
*     XMNN(I,J)=0.00000000000000
* 20  CONTINUE
*10   CONTINUE
      XMNN     =ML
*     XMNN(1,2)=MX
*     XMNN(2,1)=MX
*     XMNN(2,2)=XXI+YYI
*     DO 30 I=1,2
*     DO 40 J=1,2
      XMNN     =XMNN       + XMQQP      *A*MU
*  40 CONTINUE
* 30  CONTINUE

*     WRITE(*,*)XMQQP
```

```
      RETURN
      END
*
*.....................................................................
*
      SUBROUTINE MATMUL(A,B,M,L,N,C)
      REAL*8 A(M,L),B(L,N),C(M,N)
      DO 10 I=1,M
      DO 20 J=1,N
      C(I,J)=0.0
      DO 30 INDEX=1,L
      C(I,J)=C(I,J) + A(I,INDEX)*B(INDEX,J)
   30 CONTINUE
   20 CONTINUE
   10 CONTINUE
      RETURN
      END
*
*.....................................................................
*
      SUBROUTINE TRANS(A,B,M,L)
      REAL*8 A(M,L),B(L,M)
      DO 10 I=1,M
      DO 20 J=1,L
      B(J,I)=A(I,J)
   20 CONTINUE
   10 CONTINUE
      RETURN
      END
*
*.....................................................................
*
      SUBROUTINE TRACE(A,M,TRAC)
      REAL*8 A(M,M),TRAC
      TRAC=0.0
      DO 10 I=1,M
      TRAC=TRAC + A(I,I)
   10 CONTINUE
      RETURN
      END
*
*     MATRIX ADDITION SUBROUTINE
*
      SUBROUTINE MATADD(A,B,M,L,C)
      REAL*8 A(M,L),B(M,L),C(M,L)
      DO 10 I=1,M
      DO 20 J=1,L
      C(I,J)=A(I,J) + B(I,J)
   20 CONTINUE
   10 CONTINUE
      RETURN
      END
*
```

```
*.....................................................................
*
      SUBROUTINE BIGFOR(BIGM,BIGF,MQQ,XMQN,FQ,XMNQ,XMNN,XKN,FN,U,
     #DEFMD)
      REAL*8 BIGM(2,2),BIGF(2,1),XMQN     ,XMNQ     ,XMNN     ,XKN
      REAL*8 FN     ,U     ,MQQ,P     ,FQ ,DEFMD
      M=2
      L=1
      BIGM(1,1)=MQQ
      BIGM(1,2)=XMQN
*     BIGM(1,3)=XMQN(1,2)
      BIGM(2,1)=XMNQ
      BIGM(2,2)=XMNN
*     BIGM(2,3)=XMNN(1,2)
*     BIGM(3,1)=XMNQ(2,1)
*     BIGM(3,2)=XMNN(2,1)
*     BIGM(3,3)=XMNN(2,2)
      BIGF(1,1)=FQ
      CALL MATMUL(XKN,U,L,L,L,P)
      BIGF(2,1)=FN     -P -14.4*DEFMD
      RETURN
      END
*
*.....................................................................
*
      SUBROUTINE XLEQ(BIGM,BIGF,SOL)
      REAL*8 BIGM(2,2),BIGF(2,1),SOL(2),WKAREA(18)
      M=1
      N=2
      CALL LEQT2F (BIGM,M,N,N,BIGF,M,WKAREA,IER)
      DO 10 I=1,2
      SOL(I)=BIGF(I,1)
  10  CONTINUE
      RETURN
      END
*
*.....................................................................
*
*     SUBROUTINE GLOB(GPOS,W,DEFM)
*     REAL*8 GPOS(3),W(3,3),DEFM,RL(3)
*     RL(1)=1.0D0
*     RL(2)=0.0D0
*     RL(3)=DEFM
*     N=3
*     L=1
*     CALL MATMUL(W,RL,N,N,L,GPOS)
*     RETURN
*     END
*
      SUBROUTINE CCO(XMQQ,XMQN,XMNN,XMNQ,FTT,FN,XKN,DEFM,CTB,FCO,FC1,
     #DEFMD)
       REAL*8 XMQQ,XMNN,XMQN,FTT,FN,XKN,U,CTB,FCO,CTA,B11,B22
       REAL*8 B33,B44,F11,F22,DEFMD,F33
         B11=XMQQ/0.9985
         B22=XMQN-B11
```

```
      B33=XMNQ/0.9985
      B44=XMNN-B33
      CTA=B22/B44
      CTB=XMQQ-CTA*XMNQ
      F11=CTA*FN
      F22=(CTA*XKN)*DEFM
      F33=(CTA* 14.4*DEFMD )
      FCO=F11-F22-FTT -F33
      FC1=F11-F22-FTT
      RETURN
      END
*.....................................................................
*
*   SUBROUTINE LEQT2F (IMSL)
*
*   PURPOSE              - LINEAR EQUATION SOLUTION - FULL STORAGE
*                            MODE - HIGH ACCURACY SOLUTION
*
*   USAGE                - CALL LEQT2F (A,M,N,IA,B,IDGT,WKAREA,IER)
*
*   ARGUMENTS    A       - INPUT MATRIX OF DIMENSION N BY N CONTAINING
*                            THE COEFFICIENT MATRIX OF THE EQUATION
*                            AX = B.
*                M       - NUMBER OF RIGHT-HAND SIDES. (INPUT)
*                N       - ORDER OF A AND NUMBER OF ROWS IN B. (INPUT)
*                IA      - ROW DIMENSION OF A AND B EXACTLY AS SPECIFIED
*                            IN THE DIMENSION STATEMENT IN THE CALLING
*                            PROGRAM. (INPUT)
*                B       - INPUT MATRIX OF DIMENSION N BY M CONTAINING
*                            THE RIGHT-HAND SIDES OF THE EQUATION AX = B.
*                            ON OUTPUT, THE N BY M MATRIX OF SOLUTIONS
*                            REPLACES B.
*                IDGT    - INPUT OPTION.
*                          IF IDGT IS GREATER THAN 0, THE ELEMENTS OF
*                            A AND B ARE ASSUMED TO BE CORRECT TO IDGT
*                            DECIMAL DIGITS AND THE ROUTINE PERFORMS
*                            AN ACCURACY TEST.
*                          IF IDGT EQUALS 0, THE ACCURACY TEST IS
*                            BYPASSED.
*                          ON OUTPUT, IDGT CONTAINS THE APPROXIMATE
*                            NUMBER OF DIGITS IN THE ANSWER WHICH
*                            WERE UNCHANGED AFTER IMPROVEMENT.
*                WKAREA  - WORK AREA OF DIMENSION GREATER THAN OR EQUAL
*                            TO N**2+3N.
*                IER     - ERROR PARAMETER. (OUTPUT)
*                          WARNING ERROR
*                            IER = 34 INDICATES THAT THE ACCURACY TEST
*                              FAILED. THE COMPUTED SOLUTION MAY BE IN
*                              ERROR BY MORE THAN CAN BE ACCOUNTED FOR
*                              BY THE UNCERTAINTY OF THE DATA. THIS
*                              WARNING CAN BE PRODUCED ONLY IF IDGT IS
*                              GREATER THAN 0 ON INPUT. (SEE THE
*                              CHAPTER L PRELUDE FOR FURTHER DISCUSSION.)
*                          TERMINAL ERROR
*                            IER = 129 INDICATES THAT THE MATRIX IS
```

```
*                              ALGORITHMICALLY SINGULAR. (SEE THE
*                              CHAPTER L PRELUDE).
*                            IER = 131 INDICATES THAT THE MATRIX IS TOO
*                              ILL-CONDITIONED FOR ITERATIVE IMPROVEMENT
*                              TO BE EFFECTIVE.
*   REQD. IMSL ROUTINES - SINGLE/LUDATN,LUELMN,LUREFN,UERTST,UGETIO
*                       - DOUBLE/LUDATN,LUELMN,LUREFN,UERTST,UGETIO,
*                           VXADD,VXMUL,VXSTO
*
      SUBROUTINE LEQT2F (A,M,N,IA,B,IDGT,WKAREA,IER)
*
      DIMENSION          A(IA,1),B(IA,1),WKAREA(1)
      DOUBLE PRECISION   A,B,WKAREA,D1,D2,WA
*                                FIRST EXECUTABLE STATEMENT
*                                INITIALIZE IER
      IER=0
      JER=0
      J = N*N+1
      K = J+N
      MM = K+N
      KK = 0
      MM1 = MM-1
      JJ=1
      DO 5 L=1,N
         DO 5 I=1,N
            WKAREA(JJ)=A(I,L)
            JJ=JJ+1
    5 CONTINUE
*                                DECOMPOSE A
      CALL LUDATN (WKAREA,N,N,A,IA,IDGT,D1,D2,WKAREA(J),WKAREA(K),
     *             WA,IER)
      IF (IER.GT.128) GO TO 25
      IF (IDGT .EQ. 0 .OR. IER .NE. 0) KK = 1
      DO 15 I = 1,M
*                                PERFORMS THE ELIMINATION PART OF
*                                AX = B
         CALL LUELMN (A,IA,N,B(1,I),WKAREA(J),WKAREA(MM))
*                                REFINEMENT OF SOLUTION TO AX = B
         IF (KK .NE. 0)
     *   CALL LUREFN (WKAREA,N,N,A,IA,B(1,I),IDGT,WKAREA(J),WKAREA(MM),
     *                WKAREA(K),WKAREA(K),JER)
         DO 10 II=1,N
            B(II,I) = WKAREA(MM1+II)
   10    CONTINUE
         IF (JER.NE.0) GO TO 20
   15 CONTINUE
      GO TO 25
   20 IER = 131
   25 JJ=1
      DO 30 J = 1,N
         DO 30 I = 1,N
            A(I,J)=WKAREA(JJ)
            JJ=JJ+1
   30 CONTINUE
      IF (IER .EQ. 0) GO TO 9005
```

```
 9000 CONTINUE
      CALL UERTST (IER,6HLEQT2F)
 9005 RETURN
      END
*
*     .....................................................................
*
*        SUBROUTINE DQG4 (IMSL SUBROUTINE)
*
*        PURPOSE
*           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)
*
*        USAGE
*           CALL DQG4 (XL,XU,FCT,Y)
*           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
*        DESCRIPTION OF PARAMETERS
*           XL     - DOUBLE PRECISION LOWER BOUND OF THE INTERVAL.
*           XU     - DOUBLE PRECISION UPPER BOUND OF THE INTERVAL.
*           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION
*                    SUBPROGRAM USED.
*           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.
*        METHOD
*           EVALUATION IS DONE BY MEANS OF 4-POINT GAUSS QUADRATURE
*           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 7
*           EXACTLY. FOR REFERENCE, SEE
*           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
*           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-340.
*
      SUBROUTINE DQG4(XL,XU,FCT,Y)
*
      DOUBLE PRECISION XL,XU,Y,A,B,C,FCT

*     WRITE(*,*)XL,XU,Y

      A=.5D0*(XU+XL)
      B=XU-XL
      C=.43056815579702629D0*B
      Y=.17392742256872693D0*(FCT(A+C)+FCT(A-C))
      C=.16999052179242813D0*B
      Y=B*(Y+.32607257743127307D0*(FCT(A+C)+FCT(A-C)))
      RETURN
      END
*
*     SUBROUTINE  TRAIN(POS,LL,THICK,STRANE)
*     REAL*8 POS(2),LL,THICK(1),STRANE(1),X
*     REAL*8 C1,A1P,BETA1
*     COMMON/FCDATA/C1,A1P,BETA1
*     X=-LL/2.D0
*     STRANE(1)=THICK(1)*((POS(2) *((C1*BETA1*BETA1*(A1P*(-COS
*    # (BETA1*X)+COSH(BETA1*X))+(-SIN(BETA1*X)+SINH(BETA1*X)))))))
*     RETURN
**    END
** ***************************
         SUBROUTINE DESCU(DETO,CTB,TAG1,TAG,FCO,GKV,GKP,DEA)
          REAL*8  DETO1,DETO2,DETO,GKV,GKP,DEA,TAG1,TAG,FCO,CTB
```

```
DETO1=(-GKV*TAG1)+GKP*(DEA-TAG)
DETO2=CTB*DETO1
DETO=DETO2+FCO
RETURN
END
```

APPENDIX B

POINT TO POINT CONTROL PROGRAM

```
C This program is written to establish point to point control
C for the single-link flexible arm.

$INCLUDE:'ATLDEFS.FOR'
$INCLUDE:'ATLERRS.FOR'
      REAL*8VL,N,PIE2,PHID,L,PS,VT,BE,DM,CIM,PHIOUT,
     / PD,V3,VA(1001),
     /EFF,THETA,PH1,IK,KK,PLOUT,PL,U,I,K,PL1,U,
     /IA(1001),PA(1001),PD1,
     /EPS,DPH1,S1,V1,PC,PH1DD1,V2,PH12,THD2,PDOUT(1001),
     /PDOUT,TOR(1000),T,PDSUM
      REAL*4 RATE,D1,D2,KP,KV,HK
      INTEGER*2ADGAINS(16),ADCHAN(16),ICONFIG(16),
     /CDEV1D,CDEVFLG,SCAN,DAVAL,DAVAL2,Y,STATUS,
     /PH1DD,DA(1000),BASEADR
      COMMON/CONFIG/ICONFIG
      EQUIVALENCE (ICONFIG(KCBASEADR),BASEADR),
     /(ICONFIG(KCDEV1D),CDEV1D)
   /,(ICONFIG(KCDEVFLAGS),CDEVFLG),
     /(ICONFIG(KCSCAN),SCAN),(ICONFIG(KCCHANNELS),CHAN)
C The variables are defined in the simulation program FIOM
C except as noted below.
      RATE=50.000
      L=0.99850000
      PS=1.37888800/
      VT=3.0512/D-04
      BE=690.0006
      DM=6.2271D-05
      CIM=3.7064//2D-13
      J2=1
      EFF=0.90D0
C PDOUT is a dummy variable used in the digital filter
      PDOUT=0.0D0
      K=2.402963D-09
C THD2 is a dummy variable used to compute large angle velocity.
      THD2=0.0D0
      HK=867.24828
      KP=845./60/
      KV=60.0000
C PDSUM is the value of the filtered small angle signal.
      PDSUM=0.0D0
      PH12=-0.06102134500
      Y=2048
C PL1 is the initial pressure load drop
      PL1=23.89303D0/(DM*EFF)
      PIE2=2.00D0*DATAN(1.0D0)
      OPEN(UNIT=15,FILE='THE820F.DAT',STATUS='NEW')
C INITIALIZE THE BOARD
      STATUS=ALINIT()
      STATUS=ALSB(1)
      STATUS=ALSF(RATE)
      STATUS=ALRSET()
      STATUS=ALGC(ICONFIG)
      STATUS=ALDV(0,Y)
```

```
      J1=5
C IDENTIFY THE CONTROL PARAMETERS
*     WRITE(*,*) 'INPUT TRANSFER FUNCTION (HK)'
*     READ*,HK
*     WRITE(*,*) 'INPUT GAINS (KP,KV) (REAL)'
*     READ*,KP,KV
      WRITE(*,*) 'INPUT THE DESIRED ARM POSITION IN DEGREES'
      READ*,PHIDD1
      PRINT*,PHIDD1
      PHID=((2*PIE2/180)*PHIDD1)
      PRINT*,PHID
C COMMENCE POSITIONING OF THE ARM
      S1=10.0D0/2.048D03
      HK=HK*PHID
      U=0D0
      J1=1
C Begin data acquisition process
05    STATUS=ALAV(1,1,DAVAL)
      STATUS=ALAV(2,1,DAVAL2)
      D1=FLOAT(DAVAL-2048)
      D2=FLOAT(DAVAL2-2048)
      PDSUM=0.941/50*PDSUM + .0291290*(PDOUT + D2)
      V1=S1*D1
      V2=S1*PDSUM
      THETA=.6108652D0*V1
      THDOT=(THETA-THD2)*RATE + U/(2*RATE)
      VL=-0.4093300D0*V2/L -.001654/L
      PH1=THETA + VL
      PH1DOT=(PH1-PH12)*RATE + U/(2*RATE)
      U=HK-KP*PH1-KV*PH1DOT
      TR=U +28.4392
      FL=TR/(EFF*DM)
      FLDOT=RATE*(FL-FL1)
      Q=DM*THDOT+CTM*FL+(V1*FLDOT)/(4.0D0*BE)
      FD=FS-FL
C This section is used to install a ceiling on the value of
C current which can seen by actuator controller.
      IF(FD .LE. 5.0D-03) THEN
      I=4.0D-03
      ELSE
      I=(Q/(K*DSQRT(FD)))/1000.00
      ENDIF
      IF( I .GT. 4.0D-03) I=4.0D-03
      IF(I .LT. -4.0D-03) I=-4.0D-03
      V3=500*I
      Y=INT(2.046D03*V3/10D0 + 2048)
      STATUS=ALDV(0,Y)
      IF(J .EQ. J1) THEN
      J2=J2+1
      PA(J2)=PH1
      VA(J2)=VL
      TA(J2)=THETA
*     TOR(J2)=PDSUM
      J1=J1+5
```

```
      ENDIF
      PL1=PL
      J=J+1
      PH12=PH1
      THD2=THETA
      PDOUT=D2
      IF(J .LE. 2000) GOTO 05
10    Y=2048
      STATUS=ALDV(0,Y)
      STATUS=ALTERM()
      DO 20 J1=1,400
      WRITE(15,25)(J1*.0250),PA(J1),TA(J1),VA(J1)
25    FORMAT(1X,4(2X,F10.4))
20    CONTINUE
      END
```

APPENDIX C

TRAJECTORY CONTROL PROGRAM

```
C TRAJ.FOR IS A SAMPLE PROGRAM USED DURING TRAJECTORY
C CONTROL.

$INCLUDE:'AILDEFS.FOR'
$INCLUDE:'AILERRS.FOR'

      REAL*8VL,N,P1E2,PH1D,L,PS,V1,BE,DM,CTM,PH1DOT,
     /PD,V3,VA(1001),
     /EFF,THETA,PH1,TK,KK,PLDOT,PL,Q,I,K,PL1,U,
     /TA(1001),PA(1001),PD1,
     /EPS,DPH1,S1,V1,PC,PH1DD1,V2,PH12,THD2,PDDOT(1001),
     /PDOUT,TOR(1000),T,PDSUM
      REAL*4 RATE,D1,D2,KP,KV,HK
      INTEGER*2 ADGAINS(16),ADCHAN(16),ICONFIG(16),
     /BASEADR,CDEV1D,CDEVFLG,SCAN,DAVAL,DAVAL2,Y,
     /STATUS,PH1DD,DA(1000)
      COMMON/CONFIG/ICONFIG
      EQUIVALENCE ICONFIG(KCBASEADR),BASEADR),
     /(ICONFIG(KCDEV1D),CDEV1D)
     /(CONFIG(KCDEVFLAGS),CDEVFLG),
     /(ICONFIG(KCSCAN),SCAN),(ICONFIG(KCCHANNELS),CHAN)
C The variables are defined in the simulation program FROM
C except as noted below.
      RATE=50.000
      L=0.99850000
      PS=1.3/8888D0/
      V1=3.0512/D-04
      BE=690.0D06
      DM=6.22/1D-05
      CTM=3./064//2D-13
      J2=1
      PDDOT(J2)=0.0D0
      EFF=0.90D0
C PDOUT is a dummy variable used in the digital filter.
      PDOUT=0.0D0
      K=2.402963D-09
C THD2 is a dummy variable used to compute large angle
C velocity.
      THD2=0.0D0
      PD1=0.0D0
      HK=867.24828
      KP=845./60/
      KV=60.0000
C PDSUM is the value of the filtered small angle signal.
      PDSUM=0.0D0
      PH12=-0.06102134SD0
      Y=2048
C PD1 is the initial load pressure drop.
      PL1=23.89303D0/(DM*EFF)
      P1E2=2.00D0*DATAN(1.0D0)
      OPEN(UNIT=15,FILE='THE820G.DAT',STATUS='NEW')
C INITIALIZE THE BOARD
      STATUS=ALINIT()
      STATUS=ALSB(1)
```

```
      STATUS=ALSR(RATE)
      STATUS=ALRSET()
      STATUS=ALGC(ICONFIG)
      STATUS=ALDV(0,Y)
      J1=5
      J=0
C IDENTIFY THE CONTROL PARAMETERS
*     WRITE(*,*) 'INPUT TRANSFER FUNCTION (HK)'
*     READ*,HK
*     WRITE(*,*) 'INPUT GAINS (KP,KV) (REAL)'
*     READ*,KP,KV
*     WRITE(*,*) 'INPUT THE DESIRED ARM POSITION IN DEGREES'
*     READ*,PHIDD1
*     PRINT*,PHIDD1
*     PHID=((2*PIE2/180)*PHIDD1)
*     PRINT*,PHID
      U=0D0
      J1=1
      PD1=0D0
      S1=10D0/2.048D03
C COMMENCE POSITIONING OF THE ARM
05    IF(J .LT. 400) PHID=(1.06102/2)*(J*.005)
      IF(J .GE. 400 .AND. J .LT. 600) PHID=1.000
      IF(J .GE. 600 .AND. J .LT. 1000)
      PHID=(-1.00/2)*((J-600)*.005)+1.0
      IF(J .GE. 1000) PHID=0.00
      RK=HK*PHID
C Begin data acquisition process.
      STATUS=ALAV(1,1,DAVAL)
      STATUS=ALAV(2,1,DAVAL2)
      D1=FLOAT(DAVAL-2048)
      D2=FLOAT(DAVAL2-2048)
      PDSUM=0.941/50*PDSUM + .0291290*(PDOUT + D2)
      V1=S1*D1
      V2=S1*PDSUM
      THETA=.6108652D0*V1
      THDOT=(THETA-THD2)*RATE + U/(2*RATE)
      VL=-0.4093380D0*V2/L -.001654/L
*     VL=0.0
      PH1=THETA + VL
      PHIDOT=(PH1-PH12)*RATE + U/(2*RATE)
      U=RK-KP*PH1-KV*PHIDOT
      TR=U +28.4392
      PL=TR/(EFF*DM)
      PLDOT=RATE*(PL-PL1)
      Q=DM*THDOT+CTM*PL+(V1*PLDOT)/(4.0D0*BE)
      PD=PS-PL
C This section is used to install a ceiling on the value
C of current which can be seen by the actuator controller.
      IF(PD .LE. 5.0D-03) THEN
      I=4.0D-03
      ELSE
      I=(U/(K*DSQRT(PD)))/1000.00
      ENDIF
```

```
      IF( I .GT. 4.0D-03) I=4.0D-03
      IF(I .LT. -4.0D-03) I=-4.0D-03
      V3=500*I
      Y=INT(2.046D03*V3/10D0 + 2048)
      STATUS=ALDV(0,Y)
      IF(J .EQ. J1) THEN
      J2=J2+1
      PA(J2)=PHI
      VA(J2)=VL
      TA(J2)=THETA
*     TOR(J2)=PDSUM
*     DA(J2)=D2
      J1=J1+5
      ENDIF
      PL1=PL
      J=J+1
      PHI2=PHI
      THD2=THETA
      PDOUT=D2
      IF(J .LE. 2000) GOTO 05
10    Y=2048
      STATUS=ALDV(0,Y)
      STATUS=ALTERM()
      DO 20 J1=1,400
      WRITE(15,25)(J1*.0250),PA(J1),TA(J1),VA(J1)
25    FORMAT(1X,4(2X,F10.4))
20    CONTINUE
      END
```

LIST OF REFERENCES

1. Whitney, D. E., Book, W. J., and Lynch P. M., "Design and Control Configurations for Industrial and Space Manipulators." Proc. JACC, 1974.

2. Petroka, R. P., "Computer Simulation and Experimental Validation of a Dynamic Model (Equivalent Rigid Link System) on a Single Link Flexible Manipulator." Master's Thesis, Naval Postgraduate, Monterey, California, October 1982.

3. Gannon, K. P., "Modelling and Experimental Validation of a Single Link Flexible Manipulator." Master's Thesis, Naval Postgraduate School, Monterey, California, December 1986.

4. Park, K. S., "Control System Simulation for a Single Link Flexible Arm.", Master's Thesis, Naval Postgraduate School, Monterey, California, September 1987.

5. MOOG INC., MOOG SERIES 760 TWO STAGE FLOW CONTROL SERO-VALVE CATALOG, MOOG INC., East Aurora, New York

6. Merrit, H. E., Hydraulic Control Systems, Wiley, New York, 1967.

7. Åstrom, K. J., Wittenmark, B., Computer Controlled Systems-Theory and Design, Prentice Hall, New Jersey, 1984.

8. Franklin, G. F., and Powell, J. D., Digital Control of Dynamic System, Addison-Wesley, Phillippines, 1981.

INITIAL DISTRIBUTION LIST

		No Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5000	2
3.	Department Chairman, Code 69 Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
4.	Professor L. W. Chang, Code 69Ck Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	6
5.	Professor R. H. Nunn, Code 69Nn Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
6.	Professor D. L. Smith, Cde 69Sm Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
7.	Professor G. J. Thaler, Code 62Tr Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	1
8.	Professor J. Burl, Code 62Bl Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	1

9.	Mr. B. Foor Naval Air Engineering Center Lakehurst, New Jersey 08733	1
10.	Dr. A. L. Meyrowitz Chief of Naval Research ONR Code 433 Arlington, Virginia 22217-5000	1
11.	LT Michael Kirkland, USN Supervisor Shipbuilding Conversion and Repair Newport News, Virginia 23607-2785	2
12.	LCDR R. P. Petroka 89 Spring Street Brunswick, Maine 04011	1
13.	Professor J. F. Hamilton School of Mechanical Engineering Purdue University W. Lafayette, Indiana 47907	1

Thesis
K515 Kirkland
c.1 Implementation of dynamic control of a single-link flexible arm using a government micro-computer.